Implementation of 2.4GHz Wifi Protocol on an SDR Kit

Dissertation submitted to
Visvesvaraya National Institute of Technology, Nagpur
In partial fulfillment of the requirements for the award of the degree

Master of Technology
in
VLSI Design

by

SHIRAMSHETTY JITESH NATH
(MT23MVD004)

Under the guidance of
Dr. VASU PULIJALA

Center for VLSI and Nanotechnology
Visvesvaraya National Institute of Technology
Nagpur 440010 (India)
June 2025

Implementation of 2.4GHz Wifi Protocol on an SDR Kit

Dissertation submitted to
Visvesvaraya National Institute of Technology, Nagpur
In partial fulfillment of the requirements for the award of the degree

Master of Technology
in
VLSI Design

by

SHIRAMSHETTY JITESH NATH
(MT23MVD004)

Under the guidance of
Dr. VASU PULIJALA

Center for VLSI and Nanotechnology
Visvesvaraya National Institute of Technology
Nagpur 440010 (India)
June 2025

©Visvesvaraya National Institute of Technology, Nagpur 2025

Center for VLSI and Nanotechnology

Visvesvaraya National Institute of Technology, Nagpur

DECLARATION

I, **Shiramshetty Jitesh Nath**, hereby declare that this dissertation titled "**Implementation of 2.4GHz Wifi Protocol on an SDR Kit**" is carried out by me in the **Center for VLSI and Nanotechnology** of Visvesvaraya National Institute of Technology, Nagpur. The work is original and has not been submitted earlier whole or in part for the award of any degree/diploma at this or any other Institution / University.

Shiramshetty Jitesh Nath
Date:

CERTIFICATE

This is to certify that the dissertation titled **"Implementation of 2.4GHz Wifi Protocol on an SDR Kit "** is submitted by **Shiramshetty Jitesh Nath(MT23MVD004)** in partial fulfillment of the requirements for the award of **Master of Technology in VLSI Design,** VNIT Nagpur. The work is comprehensive, complete, and fit for final evaluation.

Dr. S. B. Dhok
Head, Center for VLSI and Technology
VNIT, Nagpur

Dr. P. Vasu
Centre for VLSI and Nanotechnology
VNIT, Nagpur.

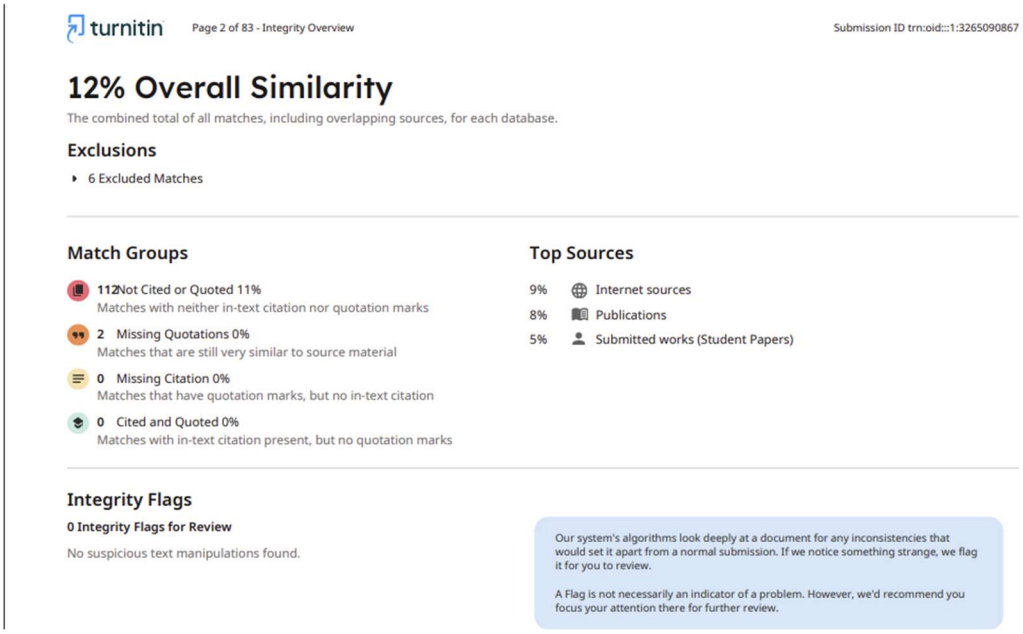
Date:

Center for VLSI and Nanotechnology

Visvesvaraya National Institute of Technology, Nagpur

NON-PLAGIARISM CERTIFICATE

Certified that M. Tech Dissertation titled **"Implementation of 2.4GHz Wifi Protocol on an SDR Kit"** is submitted by **Shiramshetty Jitesh Nath (MT23MVD004)** and has been checked for plagiarism using TURNITIN software and the overlap is found within prescribed limits. A summary of the report is given below,


turnitin Page 2 of 83 - Integrity Overview Submission ID trn:oid:::1:3265090867

12% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Exclusions

▸ 6 Excluded Matches

Match Groups

112 Not Cited or Quoted 11%
Matches with neither in-text citation nor quotation marks

2 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

9% Internet sources
8% Publications
5% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Figure 0: Plagiarism Certificate from TURNITIN Software

Shiramshetty Jitesh Nath

Dr. P. Vasu
Supervisor

Date:

Acknowledgment

This research is the result of the support and guidance of many individuals, even though only my name appears on the cover. I am sincerely grateful to everyone who contributed to making this work possible.

First and foremost, I extend my heartfelt thanks to my advisor, Dr. P. Vasu. I consider myself fortunate to have had a mentor who gave me the freedom to explore independently, while also offering timely guidance whenever I needed to refocus. His thoughtful feedback and constructive criticism throughout the course of my research sharpened my thinking and helped me define my direction. His patience and encouragement were invaluable during challenging times and played a vital role in the completion of this work.

I would also like to thank Dr. R.M. Patrikar, Dr. S.B. Dhok, Dr. R.B. Deshmukh, Dr. Nikhil Deep Gupta, and Dr. G.C. Patil for their continued guidance and insightful suggestions, which significantly enriched the quality of my project. I am equally thankful to my peers, whose support and camaraderie kept me motivated throughout this journey.

I would like to express my sincere gratitude to Prof. P. L. Patel, Director of the Institute, for cultivating an environment of academic excellence and innovation. I am also deeply thankful to Prof. V. R. Kalamkar, Dean (Academics), and Prof. R. V. Taiwade, Associate Dean (Academics – PG/PhD), for their steadfast support in facilitating student academic and research pursuits.

Above all, I am deeply indebted to my family. Their unwavering love, understanding, and patience have been the foundation of my strength, and I am truly grateful for their constant presence and support throughout this endeavor.

Shiramshetty Jitesh Nath

ABSTRACT

This thesis presents the design and implementation of an IEEE 802.11 wireless router using Software Defined Radio (SDR) technology, specifically leveraging the ZedBoard development platform integrated with the Analog Devices AD9361 RF transceiver. The ZedBoard, featuring a Xilinx Zynq-7000 SoC that combines ARM Cortex-A9 processors with FPGA fabric, provides a powerful platform for real-time signal processing applications. The AD9361 serves as a highly integrated RF front-end, offering dual-channel operation with frequency coverage from 70 MHz to 6 GHz, making it ideally suited for WiFi applications operating in the 2.4 GHz and 5 GHz bands.

The implementation involved comprehensive FPGA programming using Xilinx Vivado Design Suite to develop IEEE 802.11 protocol processing capabilities. A complete bitstream was generated incorporating both physical layer (PHY) and medium access control (MAC) layer functionalities. The system was successfully deployed on the ZedBoard platform, with the generated bitstream configuring the FPGA fabric for packet processing and protocol handling, establishing a fully functional wireless router.

The resulting SDR-based router successfully demonstrates IEEE 802.11 protocol implementation on reconfigurable hardware, providing a flexible and cost-effective alternative to traditional fixed-function networking equipment. Performance evaluation reveals competitive throughput and latency characteristics while offering unprecedented configurability for protocol modifications and enhancements. This work establishes a foundation for next-generation adaptive wireless networking systems and demonstrates the viability of SDR technology for practical networking applications. The reconfigurable nature of this implementation enables future development of application-specific routers tailored for specialized networking requirements, opening new possibilities for research in cognitive radio networks and targeted protocol optimization.

Keywords: IEEE 802.11, Software Defined Radio, FPGA, ZedBoard, AD9361, Wireless Router, Protocol Implementation, Reconfigurable Hardware, Xilinx Vivado, RF Transceiver

LIST OF FIGURES

Fig 2. 1 FMCOMMS2 and GNU Radio Integration 9

Fig 3. 1 OSI model and WiFi protocol stack 12
Fig 3. 2 OFDM Symbol Structure 14
Fig 3. 3 QPSK Constellation Diagram 15
Fig 3. 4 Collision Avoidance Mechanism 16
Fig 3. 5 Frame Transmission Timeline 17
Fig 3. 6 OFDM Subcarrier Allocation (52 Active Subcarriers) 18
Fig 3. 7 Legacy 802.11b Vs Thesis Implementation 20
Fig 3. 8 Modulation Scheme Comparision 21
Fig 3. 9 2.4GHz Frequency Implementation 21
Fig 3. 10 MIMO Spatial Multiplexing Architecture 22
Fig 3. 11 Frame Aggregation Mechanisms 23
Fig 3. 12 Channel Bonding Technology 24
Fig 3. 13 AD9361 Transceiver 25
Fig 3. 14 AD9361 Architecture 27
Fig 3. 15 Zedboard 30
Fig 3. 16 Memory System Architecture 31
Fig 3. 17 ARM Cortex-A9 Dual-Core Architecture 32
Fig 3. 18 Integrated Peripherals and I/O 33
Fig 3. 19 Zynq-7020 SoC Architecture 34
Fig 3. 20 Typical SDR Architecture 35
Fig 3. 21 Block Diagram of Zynq Boards + FMCOMMS2 37
Fig 3. 22 Zedboard + AD9361 41

Fig 4. 1 Zynq–AD9361 System Integration 42
Fig 4. 2 RF-to-Baseband Chain 43
Fig 4. 3 Digital IF & Config Flow 45
Fig 4. 4 Build Flow Diagram 47

Fig 4. 5 Boot Sequence Diagram 48

Fig 5. 1 SDR Integration Setup 52
Fig 5. 2 SSID in Mobile 53
Fig 5. 3 Spectrum Analyser Output 54

Fig 6. 1 V2V and VTI Infrastructure 57

LIST OF TABLES

Table 3. 1 Key Specifications 15
Table 3. 2 DCF vs PCF Comparision..... 17
Table 3. 3 Modulation Schemes and Data Rate Calculation..... 19
Table 3. 4 Performance Enhancement Specifications..... 24

Abbreviations

SDR : Software-Defined Radio

Zed: Zynq Evaluation and Development

SoC: System on Chip

AD9361: Analog Devices 9361

HDR: Hardware-Defined Radio

HLS: High-Level Synthesis

OFDM: Orthogonal Frequency Division Multiplexing

OSI: Open Systems Interconnection (model)

LLC: Logic Link Control

SIFS: Short Interframe Space

DIFS: DCF Interframe Space

TCP: Transmission Control Protocol

UDP: User Datagram Protocol

MPDU: MAC Protocol Data Unit

MSDU: MAC Service Data Unit

ACK: Acknowledgment

SSID: Service Set Identifier

BPSK: Binary Phase Shift Keying

QPSK: Quadrature Phase Shift Keying

QAM: Quadrature Amplitude Modulation

CSMA/CA: Carrier Sense Multiple Access with Collision Avoidance

DCF: Distributed Coordination Function

PCF: Point Coordination Function

IFFT: Inverse Fast Fourier Transform

DSSS: Direct Sequence Spread Spectrum

CCK: Complementary Code Keying

PLL: Phase-Locked Loop

LO: Local Oscillator

AGC: Automatic Gain Control

LNA: Low Noise Amplifier

TSMC: Taiwan Semiconductor Manufacturing Company

AXI: Advanced eXtensible Interface

IF: Intermediate Frequency

AXI SPI: AXI Serial Peripheral Interface

AXI GPIO: AXI General Purpose Input/Output

TABLE OF CONTENTS

Abstract i
List of Figures ii
List of Tables iv
Abbreviations v
1.Introduction 2
1.1 Background 2
1.2 Motivation 3
1.3 Objective 4
1.4 Thesis Organization 4
2.Literature Review 7
2.1 Historical Development of Software-Defined Radio (SDR) 7
2.2 Advantages and Challenges of SDR Implementations 7
2.3 Comparison with Traditional Hardware-Based Approaches 8
2.4 Evolution from IEEE 802.11 Legacy to 802.11n 8
2.5 Previous Research on FPGA Wi-Fi Implementations 8
2.6 GNU Radio and Its Applications 9
3.System Architecture and Theoretical Foundation 12
3.1. WiFi Protocol Fundamentals 12
3.1.1. OSI model and WiFi protocol stack 12
3.1.2. Physical Layer Specifications: OFDM and Modulation 14
3.1.3 MAC layer protocols (CSMA/CA, DCF, PCF) 16
3.2. IEEE 802.11 Standard Implementation Details 18
3.2.1. 802.11a: 5GHz OFDM implementation 18
3.2.2 802.11b DSSS/CCK Limitations 20
3.2.3. 802.11g: 2.4GHz OFDM Implementation 21
3.2.4. 802.11n: MIMO, aggregation, and enhanced features 22
3.3 Hardware Platform Analysis 25
3.3.1 Analog Devices AD9361(FMCOMMS2 - EBZ) 25
3.3.2. Xilinx Zynq Platforms (Zedboard) 29

3.3.3 Software Defined Radio Architecture 35
4.System Build and Deployment 41
4.1 Hardware Platform 41
4.2 FPGA Design and HDL Integration 42
4.2.1 FPGA Hardware Design and Integration using Vivado 42
4.2.2 Signal Path through AD9361 and FPGA – RF to Baseband 43
4.2.3 Software Interface and AD9361 Runtime Configuration 44
4.3 Software System and Control 45
4.4 System Image Preparation and Boot Process 48
5.Result and Analysis 52
6.Conclusion and Future Scope 56
6.1 Conclusion 56
6.2 Future Scope 57
References 59

1

INTRODUCTION

1.Introduction

1.1 Background

The modern telecommunications landscape demands adaptable and programmable networking infrastructure. Traditional networking equipment built on Application-Specific Integrated Circuits (ASICs) provides excellent performance but lacks flexibility for emerging applications such as cognitive radio networks, dynamic spectrum access, and protocol experimentation. This rigidity becomes problematic in research environments where protocol modifications are essential for advancing wireless communication technologies.

Field-Programmable Gate Arrays (FPGAs) have emerged as a powerful solution bridging the performance gap between software implementations and dedicated hardware. Modern FPGA architectures offer substantial computational resources and the ability to implement complex digital signal processing algorithms in parallel. System-on-Chip platforms integrating ARM processors with FPGA fabric create hybrid architectures combining software flexibility with hardware acceleration benefits.Wireless Local Area Networks based on 802.11 standards have become the backbone of modern connectivity, supporting applications from internet access to mission-critical industrial systems. However, heterogeneous wireless environments present challenges that conventional off-the-shelf equipment cannot adequately address. Network operators increasingly require solutions tailored to specific deployment scenarios, interference conditions, and performance requirements.

Programmable radio frequency systems have gained traction as researchers seek more intelligent and adaptive wireless networks. By implementing radio functionality in software and leveraging reconfigurable hardware platforms, networking devices can evolve with changing requirements, support multiple standards simultaneously, and incorporate advanced optimization algorithms.

Contemporary FPGA development environments provide sophisticated tools for system design and deployment. Modern synthesis flows enable complex protocol stacks to be

efficiently mapped onto reconfigurable hardware while maintaining timing closure and resource optimization. This technological convergence creates opportunities for innovative networking solutions that combine standards compliance with unprecedented configurability, addressing the growing need for adaptable wireless infrastructure in diverse application domains.

1.2 Motivation

The primary motivation for this project stems from the growing need to understand and implement wireless communication protocols at the hardware level. Traditional computer networking education relies heavily on simulation tools and commercial equipment, which limits students' exposure to the underlying signal processing and hardware implementation aspects of wireless systems. By developing an IEEE 802.11 router from scratch using FPGA technology, this project provides hands-on experience with both the theoretical foundations and practical implementation challenges of wireless networking.

Academic institutions and research laboratories often require customizable networking equipment for experimental purposes, but commercial solutions are typically closed-source and offer limited modification capabilities. This project addresses that gap by creating an open, modifiable platform that can serve as a foundation for further research and experimentation. The ability to modify protocol behavior, implement custom features, or optimize performance for specific scenarios makes this implementation valuable for educational and research purposes. Furthermore, the rapid growth of Internet of Things devices and edge computing applications creates demand for specialized networking solutions that can be optimized for specific use cases. This project demonstrates the feasibility of creating application-specific wireless infrastructure using readily available development hardware, potentially leading to more cost-effective solutions for specialized deployments where commercial equipment may be overengineered or unnecessarily expensive.

The knowledge gained from implementing a complete wireless protocol stack provides valuable insights into the trade-offs between hardware complexity, performance, and flexibility in modern communication systems.

1.3 Objective

The primary objective of this project is to successfully implement a fully functional IEEE 802.11 compliant wireless router using the ZedBoard and AD9361 SDR platform. This involves designing and synthesizing FPGA-based digital signal processing modules to handle both physical layer operations and medium access control functions. The project aims to achieve seamless integration between hardware and software components, ensuring proper packet forwarding, network interface management, and protocol compliance testing.

Key deliverables include generating optimized FPGA bitstreams that efficiently utilize available hardware resources while maintaining real-time performance requirements. The implementation must demonstrate interoperability with standard WiFi devices and validate proper protocol operation through comprehensive testing procedures. Additionally, the project seeks to establish a reproducible development methodology that documents the complete design flow from initial specifications through final deployment.

1.4 Thesis Organization

Chapter 1: Background, Motivation, and Objectives

This chapter introduces Software Defined Radio (SDR) technology and establishes the research motivation for open-source WiFi solutions. It defines the primary objectives of implementing and analyzing an open-source WiFi system and outlines the scope of the thesis.

Chapter 2: Literature Review

The literature review examines existing WiFi implementations and Software Defined Radio applications in wireless communications. It analyzes open-source wireless projects and FPGA-based systems, identifying research gaps that the proposed solution addresses.

Chapter 3: System Architecture and Theoretical Foundation

This chapter covers WiFi protocol fundamentals and IEEE 802.11 a/b/g/n specifications. It examines the AD9361 transceiver and Xilinx Zynq ZedBoard platform, concluding with system architecture and FPGA-based baseband processing design.

Chapter 4: Implementation and Development

The implementation chapter describes the complete development process including software stack implementation and FPGA design integration. It details the practical realization of the system and discusses the challenges encountered during development.

Chapter 5: Results and Performance Analysis

This chapter presents the experimental methodology and analyzes the overall system performance. It evaluates the functionality of the implemented design and validates the design objectives through testing and analysis.

Chapter 6: Conclusion and Future Scope

The concluding chapter summarizes key achievements and research contributions. It discusses implementation limitations and explores future enhancements including 802.11ac/ax support, AI integration, and long-term research directions in 6G and cognitive radio technologies.

2

LITERATURE REVIEW

2.Literature Review

2.1 Historical Development of Software-Defined Radio (SDR)

Software-Defined Radio (SDR) originated in the 1970s and 1980s through collaborations between U.S. defense organizations and private research entities. The concept was further refined in the early 1990s by Joe Mitola III, who introduced the term "software radio," envisioning radios where components traditionally implemented in hardware would be executed through software. This paradigm shift allowed for unprecedented flexibility in radio communication systems.

By the early 2000s, SDR technology had matured, leading to widespread adoption in both military and commercial sectors. The integration of SDR into various applications, including satellite communications and cellular networks, marked a significant evolution in radio technology.

2.2 Advantages and Challenges of SDR Implementations

Advantages:

Flexibility and Reconfigurability: SDRs can adapt to multiple communication standards and frequencies through software updates, eliminating the need for hardware modifications.

Cost-Effectiveness: By consolidating multiple functionalities into a single device, SDRs reduce the need for multiple hardware components, leading to cost savings in production and maintenance.

Rapid Deployment: Software updates allow for quick implementation of new features and standards, facilitating faster deployment in dynamic environments.

Challenges:

Computational Demands: Real-time signal processing requires significant computational resources, which can be a limitation in portable or power-constrained devices.

Dynamic Range Limitations: Some SDR designs exhibit poor dynamic range compared to traditional hardware radios, affecting their ability to handle weak signals amidst strong interference.

Security Concerns: The reprogrammable nature of SDRs can pose security risks if not properly managed, as unauthorized modifications could lead to vulnerabilities.

2.3 Comparison with Traditional Hardware-Based Approaches

Traditional hardware radios (HDRs) are designed with fixed functionalities, optimized for specific tasks, leading to high performance in their designated roles. However, this rigidity limits their adaptability to new standards or frequencies.

In contrast, SDRs offer a software-centric approach, enabling dynamic reconfiguration to support various communication protocols. This flexibility is particularly advantageous in environments where multiple standards coexist or evolve rapidly. While SDRs may face challenges in terms of processing power and dynamic range, their adaptability and cost-effectiveness often outweigh these limitations in many applications.

2.4 Evolution from IEEE 802.11 Legacy to 802.11n

The IEEE 802.11 standards have undergone significant evolution since their inception.

- **802.11 (1997):** The original standard, offering data rates up to 2 Mbps using the 2.4 GHz band.
- **802.11b (1999):** Introduced higher data rates up to 11 Mbps, maintaining the 2.4 GHz frequency.
- **802.11a (1999):** Operated in the 5 GHz band with data rates up to 54 Mbps, offering less interference but shorter range.
- **802.11g (2003):** Combined the best of 802.11a and b, providing 54 Mbps in the 2.4 GHz band.
- **802.11n (2009):** Marked a significant leap with data rates up to 600 Mbps, utilizing Multiple Input Multiple Output (MIMO) technology and channel bonding.

Each iteration addressed the growing demand for higher data rates, improved reliability, and better spectrum efficiency, laying the groundwork for subsequent standards like 802.11ac and 802.11ax.

2.5 Previous Research on FPGA Wi-Fi Implementations

Field-Programmable Gate Arrays (FPGAs) have been a pivotal technology in the development of Wi-Fi communication systems due to their reconfigurability, high

throughput, and suitability for parallel processing. Research in this area has primarily focused on designing efficient baseband and MAC layer modules for wireless systems that adhere to standards such as IEEE 802.11a/g/n.

Baseband Processing and High-Level Synthesis (HLS):

A key direction in recent research is the use of High-Level Synthesis (HLS) tools to streamline FPGA design processes. HLS allows developers to describe hardware in high-level languages like C/C++, significantly reducing the time and complexity involved in creating functional modules. One study effectively demonstrated the design of Orthogonal Frequency Division Multiplexing (OFDM) baseband blocks—such as synchronization, channel estimation, and equalization—using HLS. These modules were synthesized into an FPGA design that supported real-time operations at data rates consistent with IEEE 802.11n, showing minimal trade-offs in terms of latency and hardware utilization compared to traditional VHDL-based approaches.

FPGAs serve as an excellent platform for experimental and prototypical development in wireless communications.

2.6 GNU Radio and Its Applications

GNU Radio is a free, open-source toolkit that provides signal processing blocks for implementing SDRs. It supports development in Python and C++, offering a graphical interface for designing and testing radio systems. GNU Radio has been widely adopted in academic research, industry, and hobbyist communities for its flexibility and extensive library of modules.

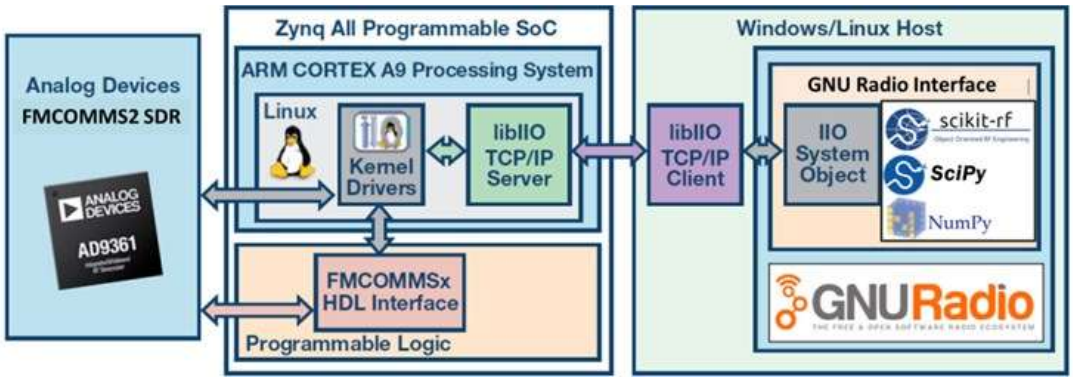


Fig 2. 1 FMCOMMS2 and GNU Radio Integration

Applications of GNU Radio include:

- **Educational Use:** Facilitating learning in digital signal processing and wireless communications.
- **Research and Development:** Prototyping new communication protocols and testing novel algorithms.
- **Spectrum Analysis:** Monitoring and analyzing radio frequency spectrum for various applications.
- **Amateur Radio:** Enabling hobbyists to experiment with and build custom radio systems.

3

SYSTEM ARCHITECTURE AND THEORETICAL FOUNDATION

3.System Architecture and Theoretical Foundation

3.1. WiFi Protocol Fundamentals

3.1.1. OSI model and WiFi protocol stack

The Open Systems Interconnection (OSI) reference model provides a seven-layer framework for understanding network communications, where each layer performs specific functions while working together to transmit data between networked devices. This hierarchical approach enables modular design and standardization across different network technologies. WiFi (IEEE 802.11) operates primarily in the Physical (Layer 1) and Data Link (Layer 2) layers of the OSI model, handling radio wave transmission and medium access control respectively.

Fig 3. 1 OSI model and WiFi protocol stack

The Physical Layer (Layer 1) in WiFi systems encompasses the actual radio frequency operations, including signal modulation, power amplification, antenna management, and channel characteristics. Modern WiFi standards utilize Orthogonal Frequency Division Multiplexing (OFDM) for efficient spectrum utilization, where data is

transmitted across multiple subcarriers simultaneously. This layer handles critical timing requirements, frequency synchronization, and automatic gain control mechanisms essential for reliable wireless communication.

The Data Link Layer (Layer 2) is subdivided into two sublayers: Logical Link Control (LLC) and Medium Access Control (MAC). The MAC sublayer implements the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol, which coordinates channel access among multiple wireless devices. This includes implementing the Distributed Coordination Function (DCF), managing frame acknowledgments, and maintaining precise timing intervals such as Short Interframe Space (SIFS) and Distributed Interframe Space (DIFS).

In traditional networking architectures, upper layers (Application through Transport) handle user applications, data formatting, session management, and end-to-end communication reliability. The Network Layer manages routing and addressing through protocols like IP, while Transport Layer ensures reliable data delivery via TCP or UDP. WiFi systems primarily focus on the lower two layers, interfacing with standard networking protocols through well-defined APIs and driver interfaces.

Our project implements a complete IEEE 802.11-compatible system by integrating Linux's mac80211 subsystem with custom FPGA-based baseband processing, creating a comprehensive software-defined radio solution that spans both hardware and software domains. The mac80211 framework provides a standardized interface between the Linux kernel networking stack and WiFi hardware drivers, handling frame management, power saving mechanisms, and regulatory compliance.

Our system architecture demonstrates how software-defined radio principles can be applied to create flexible, programmable wireless systems. By implementing time-critical MAC functions in FPGA hardware while leveraging Linux's robust networking infrastructure for higher-layer protocols, our implementation achieves both stringent real-time performance requirements and exceptional system flexibility. The FPGA handles microsecond-level timing constraints required for WiFi compliance, while the ARM processor manages complex protocol state machines and user applications.

This hybrid hardware-software approach enables researchers and developers to modify and experiment with WiFi protocols at multiple layers, from low-level physical signal processing and channel coding to high-level application functionality and network management. The modular design facilitates protocol experimentation, security research, and performance optimization, making it an invaluable platform for advancing wireless communication research and providing hands-on educational experiences in modern wireless systems.

3.1.2. Physical Layer Specifications: OFDM and Modulation

OFDM Implementation

This project implements IEEE 802.11 physical layer using 52 orthogonal subcarriers with baseband data independently modulated using quadrature amplitude modulation. The OFDM symbol structure divides the available spectrum into multiple narrow subcarriers, enabling robust transmission against multipath fading.

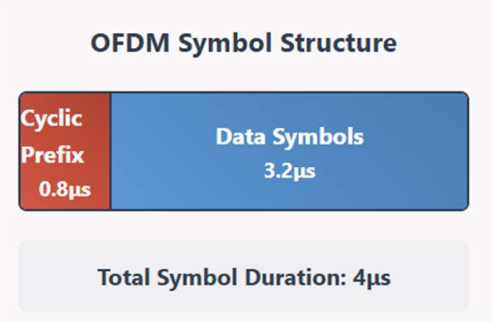


Fig 3. 2 OFDM Symbol Structure

Modulation Schemes

The system supports adaptive modulation based on channel conditions:

- **BPSK (1 bit/symbol):** Robust for poor channel conditions
- **QPSK (2 bits/symbol):** Balanced performance/throughput
- **16-QAM (4 bits/symbol):** Higher throughput for good channels
- **64-QAM (6 bits/symbol):** Maximum throughput in excellent conditions

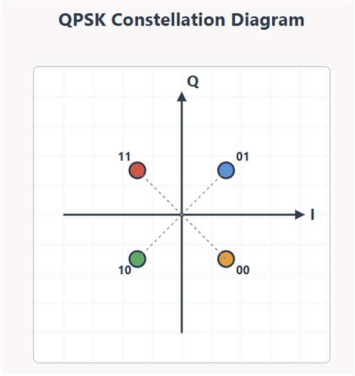


Fig 3. 3 QPSK Constellation Diagram

System Architecture

The implementation addresses Wi-Fi's strict SIFS timing requirement of 10μs/16μs acknowledgment response through FPGA-based processing, eliminating PC-based latency issues. The baseband processing chain includes the following stages:

1. **Channel Coding:** Error correction and detection
2. **Symbol Interleaving:** Data scrambling for burst error protection
3. **OFDM Modulation:** Multi-carrier signal generation
4. **RF Transmission:** Signal amplification and antenna transmission

Critical Timing: FPGA processing eliminates PC-based latency (≥100μs), enabling real-time SIFS compliance.

Parameter	Specification
Bandwidth	20MHz
Frequency Bands	2.4GHz & 5GHz
Power Control	Configurable Transmission Power
Rate Selection	Adaptive Algorithms
Subcarriers	52 Orthogonal Carriers
Symbol Duration	4us (0.8us CP + 3.2us data)

Table 3. 1 Key Specifications

The system optimizes real-time baseband operations with channel coding, interleaving, and OFDM modulation stages for seamless Wi-Fi infrastructure integration while maintaining full protocol compliance for research and development applications.

3.1.3 MAC layer protocols (CSMA/CA, DCF, PCF)

The Medium Access Control (MAC) layer in IEEE 802.11 networks implements sophisticated protocols to manage wireless channel access efficiently. The primary mechanisms include Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), Distributed Coordination Function (DCF), and Point Coordination Function (PCF), each serving distinct roles in wireless communication.

Collision Avoidance Mechanism
1. Sense Channel: Listen for ongoing transmissions before attempting to send
2. Wait DIFS: If channel busy, wait for Distributed Inter-Frame Space
3. Random Backoff: Generate random backoff time to avoid collisions
4. Transmit: Send data frame when channel becomes idle

Fig 3. 4 Collision Avoidance Mechanism

DCF vs PCF Comparison

IEEE 802.11 defines two fundamental coordination functions that operate differently to manage channel access and ensure fair medium sharing among wireless stations.

Distributed Coordination Function (DCF)	Point Coordination Function (PCF)
Contention-based access	Polling-based access
No central coordination	Centralized scheduling
Random backoff mechanism	Access Point controlled
Best-effort service	Guaranteed service periods
Mandatory implementation	Optional implementation

Table 3. 2 DCF vs PCF Comparision

MAC Timing Relationships:

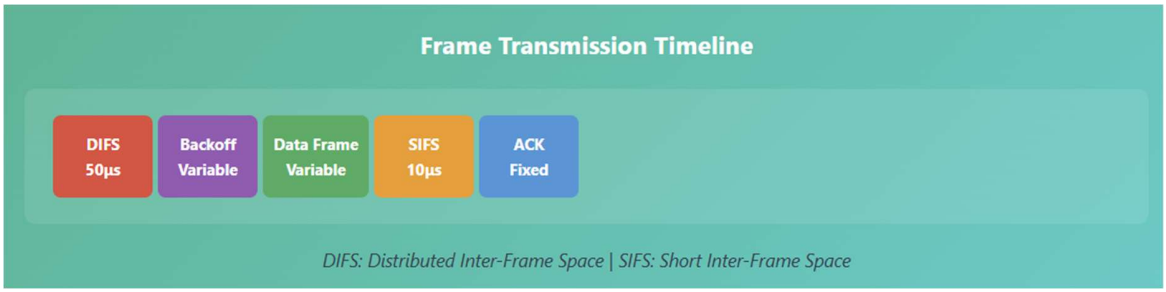


Fig 3. 5 Frame Transmission Timeline

Key Protocol Features:

- CSMA/CA prevents collisions through carrier sensing and exponential backoff algorithms.
- PCF provides deterministic access for time-sensitive applications through polling.
- DCF ensures fair channel access through distributed random-access mechanisms.

These MAC protocols work synergistically to provide robust wireless communication. DCF handles most data traffic through its distributed approach, while PCF can be implemented for applications requiring guaranteed bandwidth and low latency. The combination enables

802.11 networks to support diverse traffic types efficiently while maintaining backward compatibility and scalability across different deployment scenarios.

3.2. IEEE 802.11 Standard Implementation Details

3.2.1. 802.11a: 5GHz OFDM implementation

IEEE 802.11a revolutionized WiFi by introducing OFDM (Orthogonal Frequency Division Multiplexing) in the 5GHz band, achieving data rates up to 54 Mbps. Unlike single-carrier systems, OFDM divides the 20MHz channel into 64 orthogonal subcarriers, with 52 actually used for transmission. This parallel transmission approach provides excellent resistance to multipath fading and interference.

The system employs adaptive modulation and coding, dynamically selecting from BPSK, QPSK, 16-QAM, or 64-QAM based on channel conditions. Combined with convolutional coding rates of 1/2, 2/3, and 3/4, this creates eight different data rate modes. Operating in the less congested 5GHz spectrum provides cleaner channels and better performance than 2.4GHz systems, making 802.11a ideal for enterprise and high-density environments.

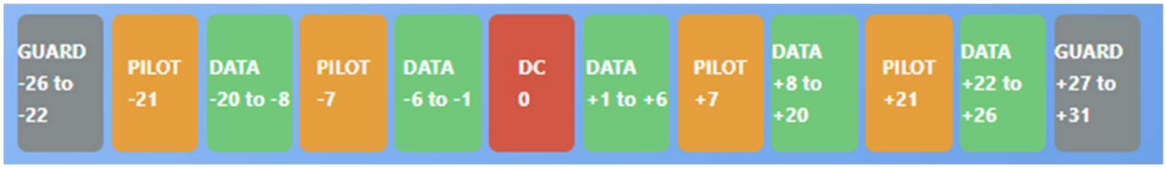


Fig 3. 6 OFDM Subcarrier Allocation (52 Active Subcarriers)

Data Subcarriers (48 green): Carry the actual information bits using various modulation schemes. These are numbered -26 to +26, excluding pilots, DC, and guard bands.

Pilot Subcarriers (4 orange): Located at positions -21, -7, +7, +21, these transmit known reference signals for phase and frequency tracking, essential for coherent demodulation at the receiver.

DC Subcarrier (1 red): At position 0 (center frequency), this is nulled to avoid DC offset issues in direct conversion receivers.

Guard Subcarriers (grey): Unused subcarriers at band edges prevent interference with adjacent channels and provide spectral shaping.

Symbol Timing Critical for OFDM

Total Symbol Duration: 4.0 μs provides 250,000 symbols per second transmission rate.

Cyclic Prefix (CP): The 0.8 μs guard interval copies the last portion of the symbol to the beginning. This prevents inter-symbol interference (ISI) from multipath reflections up to 240 meters (0.8 μs × 3×10^8 m/s). Without CP, delayed multipath signals would corrupt the next symbol.

Data Portion: The 3.2 μs interval contains the actual 64-point IFFT output. The subcarrier spacing of 312.5 kHz (20 MHz ÷ 64) ensures orthogonality - each subcarrier completes an integer number of cycles during this period.

Modulation	Coding Rate	Bits/Symbol	Coded Bits/Symbol	Data Rate(Mbps)
BPSK	1/2	1	48 × (1 ÷ 2) = 24	6
QPSK	½	2	48 × (2 ÷ 2) = 48	12
16-QAM	½	4	48 × (4 ÷ 2) = 96	24
64-QAM	3/4	6	48 × (6 × ¾) = 216	54

Table 3. 3 Modulation Schemes and Data Rate Calculation

3.2.2 802.11b DSSS/CCK Limitations

This project implements an open-source IEEE 802.11 WiFi baseband design using FPGA technology. However, it faces significant architectural limitations when handling legacy 802.11b Direct Sequence Spread Spectrum (DSSS) and Complementary Code Keying (CCK) modulation schemes.

Core Limitation: This project supports only OFDM rates, automatically converting 802.11b rates like 1Mbps to 6Mbps OFDM equivalents, fundamentally altering the original signal characteristics.

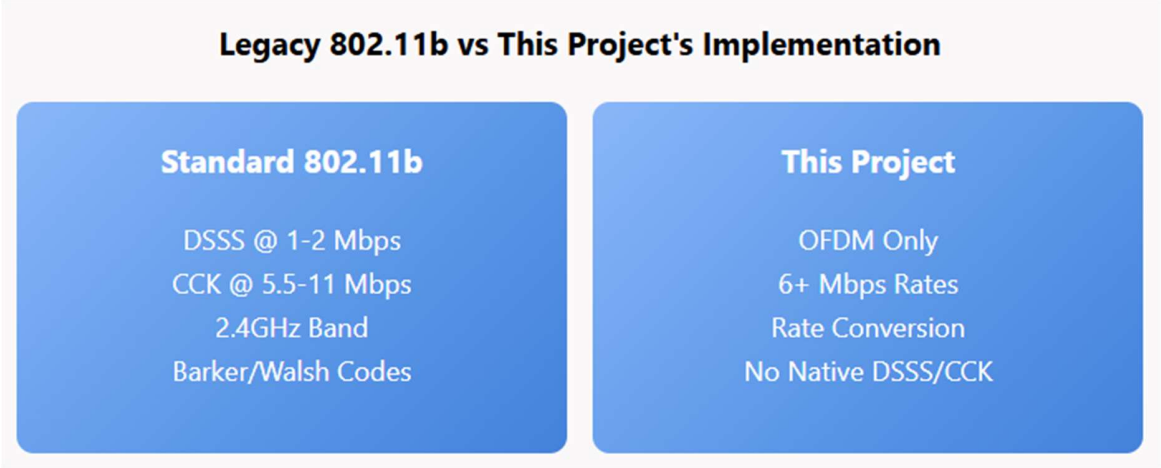


Fig 3. 7 Legacy 802.11b Vs Thesis Implementation

Frame Control Handling: The system processes FC (Frame Control) fields but converts underlying modulation schemes, potentially affecting legacy device compatibility and precise timing requirements of DSSS/CCK protocols.

The 2.4GHz DSSS/CCK limitations stem from this project's FPGA architecture being optimized for OFDM modulation rather than legacy spread spectrum techniques. While IEEE 802.11b-1999 defines CCK modulation at 5.5 and 11 Mbps with optional PBCC modes, this project's implementation bypasses these modulation schemes entirely.

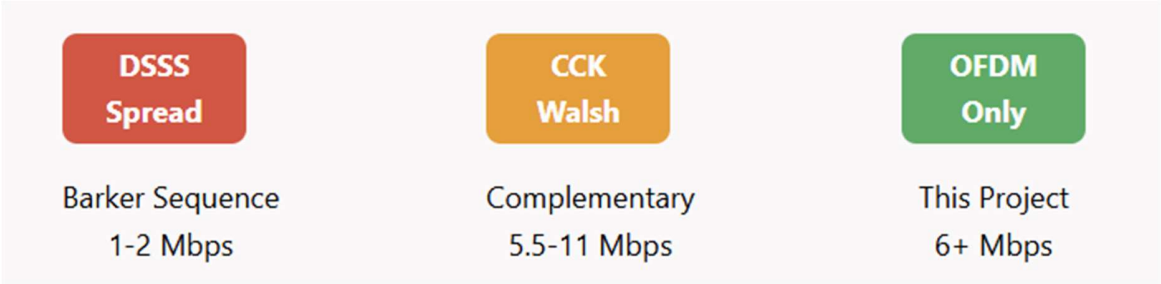


Fig 3. 8 Modulation Scheme Comparision

This architectural decision prioritizes modern OFDM efficiency over legacy compatibility, making the system unsuitable for environments requiring true 802.11b DSSS/CCK support or precise low-rate communications typical in IoT and sensor networks.

3.2.3. 802.11g: 2.4GHz OFDM Implementation

This project implements IEEE 802.11g OFDM modulation exclusively in the 2.4GHz spectrum, utilizing FPGA-based baseband processing for high-performance wireless communication. The system operates on SoftMAC architecture with hardware-accelerated OFDM processing. The OFDM part is almost similar to 5GHz

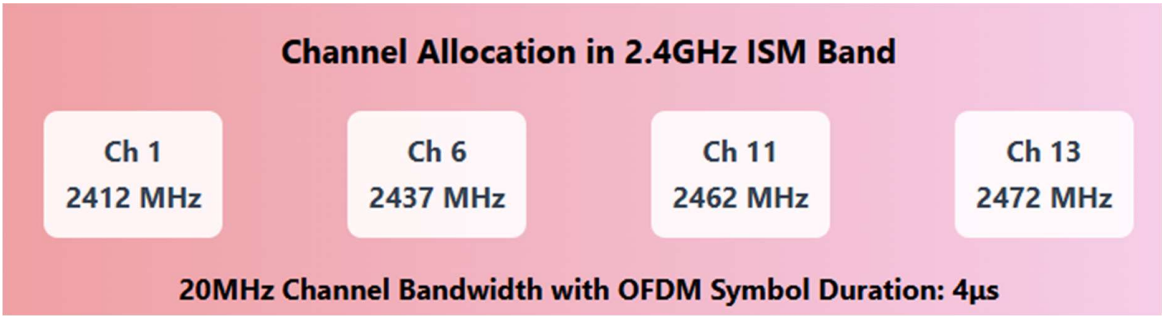


Fig 3. 9 2.4GHz Frequency Implementation

The 802.11g OFDM implementation achieves robust performance in the crowded 2.4GHz band through advanced error correction and adaptive rate selection. The system provides fully Linux-compatible WiFi functionality while maintaining real-time OFDM processing requirements essential for reliable wireless communication.

3.2.4. 802.11n: MIMO, aggregation, and enhanced features

IEEE 802.11n represents a quantum leap in wireless networking technology, introducing Multiple-Input Multiple-Output (MIMO) antenna systems, frame aggregation, and advanced channel bonding to achieve unprecedented throughput and reliability in both 2.4GHz and 5GHz bands. The 802.11n standard, ratified in 2009, fundamentally transformed wireless communication by implementing MIMO technology that utilizes multiple antennas for simultaneous data transmission. This spatial multiplexing approach enables theoretical data rates up to 600 Mbps through four spatial streams, each operating at 150 Mbps. The standard introduces channel bonding, combining two adjacent 20MHz channels into a single 40MHz channel, effectively doubling bandwidth capacity. Frame aggregation mechanisms, including A-MSDU and A-MPDU, significantly reduce protocol overhead by bundling multiple frames into single transmissions.

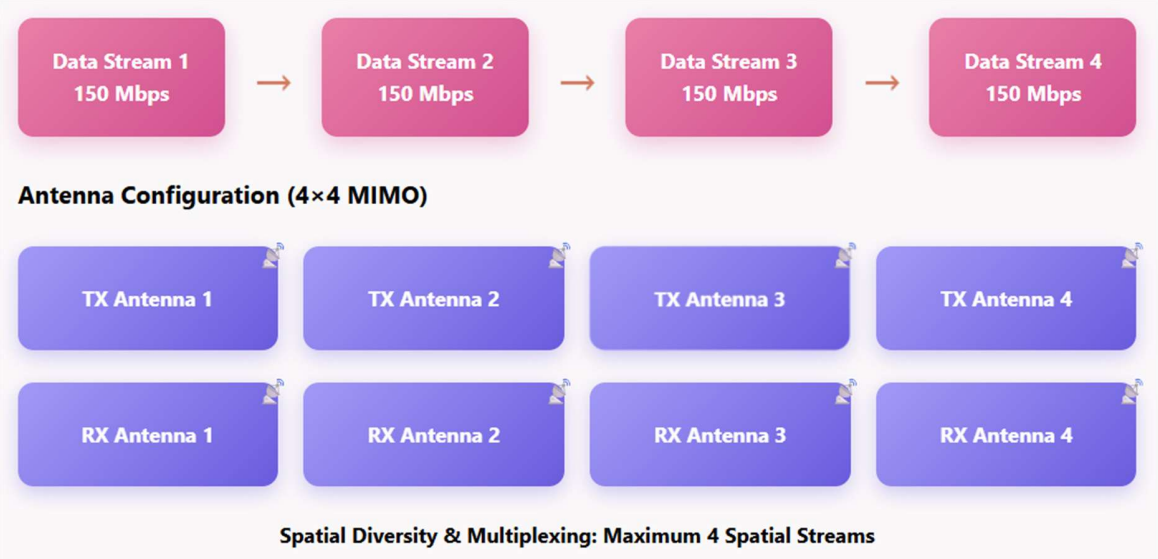


Fig 3. 10 MIMO Spatial Multiplexing Architecture

Spatial Stream Processing: The diagram illustrates how MIMO technology splits a single high-speed data stream into multiple parallel spatial streams. Each stream (150 Mbps) is processed independently and transmitted simultaneously through different antennas,

achieving spatial multiplexing where the same frequency spectrum carries multiple data streams.

Antenna Array Function: The 4×4 configuration shows four transmit antennas sending spatially distinct signals and four receive antennas capturing these signals. Each transmit antenna broadcasts a unique combination of all four data streams with different phase and amplitude coefficients, creating spatial signatures that allow the receiver to separate and reconstruct the original streams.

Channel Matrix Processing: The wireless channel between transmit and receive antenna arrays creates a 4×4 matrix where each element represents the channel response between one TX and one RX antenna. The receiver uses advanced signal processing algorithms (like Zero-Forcing or MMSE) to invert this channel matrix and extract individual spatial streams from the combined received signal.

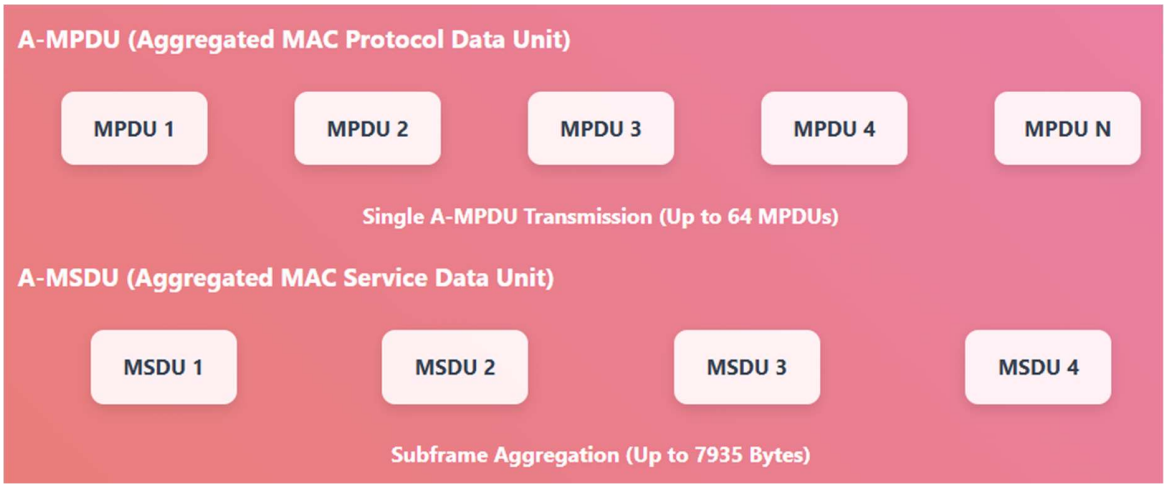


Fig 3. 11 Frame Aggregation Mechanisms

A-MPDU Mechanism: The A-MPDU diagram shows how multiple MAC Protocol Data Units (complete MAC frames with headers and FCS) are concatenated into a single physical transmission. Each MPDU maintains its individual MAC header, payload, and frame check sequence, but they're transmitted together with a single PHY header and preamble, dramatically reducing per-frame overhead.

A-MSDU Structure: The A-MSDU diagram illustrates subframe aggregation where multiple MAC Service Data Units (data payloads without individual MAC headers) are combined under a single MAC header. Each MSDU becomes a subframe with a minimal subframe header containing destination address, source address, and length field, maximizing payload efficiency.

Efficiency Gains: Traditional 802.11 protocols spend significant time on inter-frame spacing, acknowledgments, and contention for each individual frame. Aggregation reduces this overhead from approximately 50% to less than 10% by batching multiple data units. A-MPDU allows up to 64 individual MPDUs (maximum 65,535 bytes) while A-MSDU can aggregate up to 7,935 bytes of payload data, both dramatically improving channel utilization and effective throughput.

Block Acknowledgment Integration: Both aggregation types work with Block ACK mechanisms, where instead of acknowledging each frame individually, the receiver sends a single block acknowledgment indicating which frames were successfully received, further reducing protocol overhead and improving efficiency.

Configuration	Spatial Streams	Channel Width(MHz)	Max Data Rate(Mbps)
1x1 SISO	1	20	72.2
2X2 MIMO	2	20	144.4
2X2 MIMO	2	40	300
4X4 MIMO	4	40	600

Table 3. 4 Performance Enhancement Specifications

Fig 3. 12 Channel Bonding Technology

The 802.11n standard's revolutionary MIMO implementation enables spatial diversity and multiplexing, dramatically improving both throughput and reliability. Frame aggregation reduces protocol overhead from 50% to less than 10%, while 40MHz channel bonding doubles spectral efficiency. These enhancements collectively deliver up to 10× performance improvement over previous 802.11g systems, establishing 802.11n as the foundation for modern high-speed wireless networking applications requiring robust connectivity and substantial bandwidth capacity.

3.3 Hardware Platform Analysis

3.3.1 Analog Devices AD9361(FMCOMMS2 - EBZ)

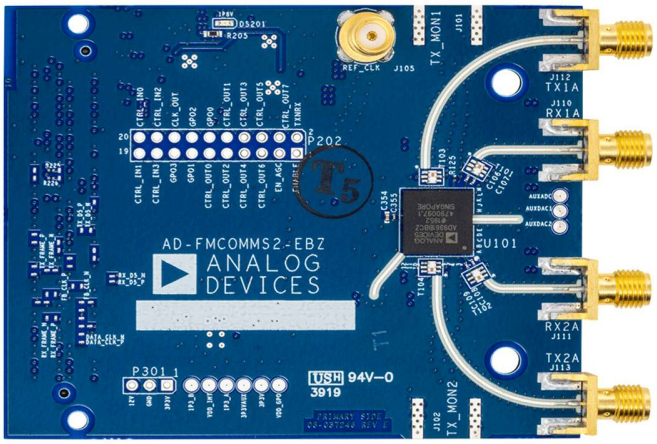


Fig 3. 13 AD9361 Transceiver

Overview

The Analog Devices AD9361 is a highly integrated radio frequency (RF) agile transceiver designed for software-defined radio (SDR) applications. This device combines transmit and receive functionality in a single chip, supporting frequency ranges from 70 MHz to 6 GHz, making it suitable for a wide variety of wireless communication standards including LTE, WCDMA, GSM, WiFi, and many others.

RF Performance Characteristics:

Frequency Range and Tuning

The AD9361 operates across an impressive frequency spectrum from 70 MHz to 6.0 GHz, achieved through its advanced fractional-N phase-locked loop (PLL) architecture. The device features two independent local oscillator (LO) chains, enabling simultaneous transmission and reception on different frequencies. The frequency resolution is exceptionally fine at 2.4 Hz steps, providing precise channel selection capabilities.

The RF front-end incorporates automatic gain control (AGC) with a dynamic range exceeding 80 dB. The receiver chain includes low-noise amplifiers (LNAs) with noise figures as low as 1.3 dB at 800 MHz, while the transmitter section features power amplifiers capable of delivering up to +7 dBm output power with excellent linearity characteristics.

Receiver Performance

The receiver architecture employs a direct-conversion topology with integrated I/Q demodulation. Key performance metrics include:

- **Sensitivity**: Better than -94 dBm for a 12 dB SINAD at 1 MHz bandwidth
- **Dynamic Range**: Greater than 80 dB with automatic gain control
- **Phase Noise**: -136 dBc/Hz at 1 MHz offset from a 2.14 GHz carrier
- **Spurious-Free Dynamic Range**: Exceeds 85 dBc in a 1 MHz bandwidth
- **Image Rejection**: Greater than 80 dB across the entire frequency range

The receiver includes sophisticated filtering capabilities with programmable bandwidth selection from 200 kHz to 56 MHz, implemented through a combination of analog and digital filtering techniques.

Transmitter Performance

The transmitter section features a direct-conversion architecture with integrated I/Q modulation. Critical specifications include:

- **Output Power**: Adjustable from -89.75 dBm to +7 dBm in 0.25 dB steps
- **Error Vector Magnitude (EVM)**: Less than -43 dB for 64-QAM OFDM signals
- **Adjacent Channel Leakage Ratio (ACLR)**: Better than -55 dBc for LTE signals
- **Spectral Mask Compliance**: Meets stringent requirements for multiple wireless standards
- **Output Return Loss**: Greater than 10 dB across the operating frequency range

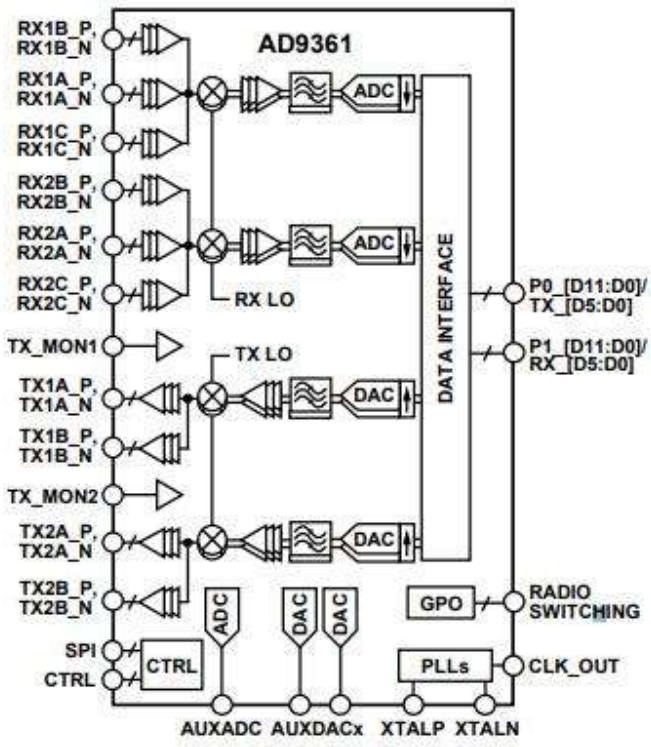


Fig 3. 14 AD9361 Architecture

Baseband Processing Features

Analog-to-Digital and Digital-to-Analog Conversion

The AD9361 incorporates high-performance 12-bit analog-to-digital converters (ADCs) and digital-to-analog converters (DACs) operating at sample rates up to 61.44 MSPS. The converters feature excellent dynamic performance with signal-to-noise ratios exceeding 74 dB and spurious-free dynamic range better than 80 dBc.

Digital Signal Processing Engine

The integrated digital signal processing (DSP) engine provides comprehensive baseband processing capabilities:

Filtering and Decimation: Configurable finite impulse response (FIR) filters with programmable coefficients enable precise channel filtering and sample rate conversion. The device supports decimation ratios from 1 to 48, allowing flexible adaptation to various signal bandwidths.

Automatic Gain Control: The digital AGC system provides fast and accurate power level adjustment with programmable attack and decay time constants. The AGC can operate in both fast and slow modes, adapting to different signal environments and requirements.

DC Offset and I/Q Imbalance Correction: Advanced calibration algorithms continuously monitor and correct for DC offset drift and I/Q amplitude and phase imbalances, ensuring optimal signal quality across temperature and process variations.

Digital Pre-Distortion: The transmitter path includes digital pre-distortion capabilities to linearize the RF power amplifier response, improving spectral efficiency and reducing adjacent channel interference.

Flexible Clocking Architecture

The AD9361 features a sophisticated clocking system with multiple programmable dividers and phase-locked loops. The device can operate with reference clock frequencies from 10 MHz to 80 MHz, with internal multiplication to generate the required sampling and LO frequencies. Clock jitter performance is exceptional, with RMS jitter typically less than 0.3 ps when using a clean reference source.

Integration Considerations

Power Management

The AD9361 incorporates sophisticated power management features, including individual power-down controls for unused circuit blocks and programmable sleep modes. Total power consumption is optimized through dynamic adjustment based on operating conditions, with typical consumption ranging from 1.3W to 2.5W depending on configuration.

Thermal Management

Proper thermal design is crucial for optimal performance. The device is available in a 10mm × 10mm, 144-ball chip-scale ball grid array (CSBGA) package with exposed thermal pad for efficient heat dissipation. Recommended operating temperature range is -40°C to +85°C for industrial applications.

Layout Considerations

Critical layout guidelines include proper power supply decoupling, controlled impedance routing for high-speed digital signals, and careful attention to ground plane

integrity. The RF sections require particular attention to minimize coupling between transmit and receive paths and ensure adequate isolation from digital switching noise.

The AD9361's comprehensive feature set, exceptional performance characteristics, and flexible interface options make it an ideal choice for modern software-defined radio applications, providing developers with a powerful platform for implementing advanced wireless communication systems across a wide range of frequency bands and modulation schemes.

3.3.2. Xilinx Zynq Platforms (Zedboard)

Introduction to Zynq SoC Architecture

The Xilinx Zynq-7000 All Programmable System-on-Chip (AP SoC) represents a revolutionary approach to embedded system design, combining the flexibility of programmable logic with the performance and software ecosystem of ARM processors. This heterogeneous architecture enables designers to partition functionality between hardware and software optimally, delivering unprecedented performance per watt for embedded applications.

The Zynq family fundamentally differs from traditional FPGA architectures by integrating a complete ARM-based processing system alongside programmable logic fabric on a single silicon die. This tight integration eliminates the need for external processors and reduces system complexity while enabling new levels of performance and functionality.

ZedBoard (Zynq-7020) Architecture Overview

The ZedBoard development platform, built around the Zynq-7020 SoC, serves as an excellent reference design showcasing the capabilities of the Zynq architecture. The Zynq-7020 features a dual-core ARM Cortex-A9 processor running at up to 866 MHz, coupled with 85,000 logic cells of programmable fabric, making it suitable for a wide range of applications from industrial automation to software-defined radio.

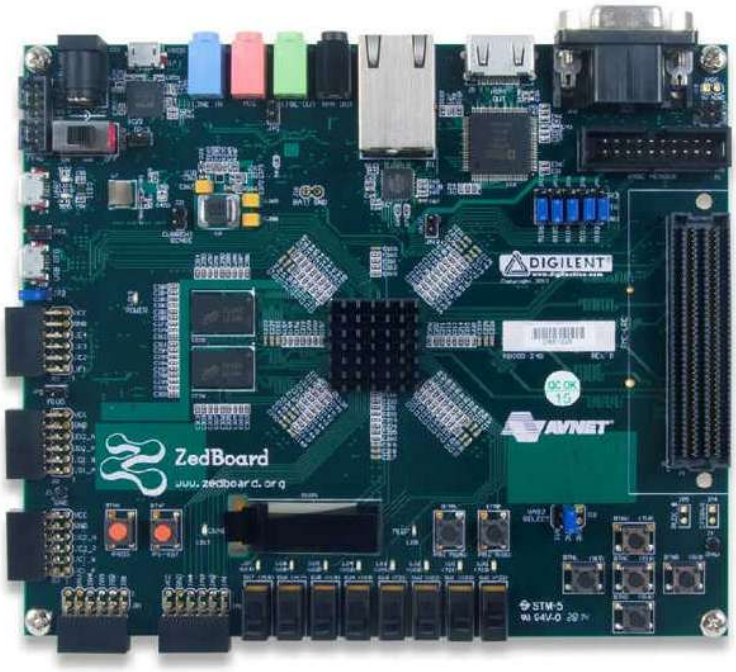


Fig 3. 15 Zedboard

Silicon Architecture Foundation

The Zynq-7020 is manufactured using TSMC's 28nm High-Performance Low-Power (HPL) process technology, enabling the integration of both processor and FPGA logic on a single die with shared power and clock distribution networks. The silicon architecture features separate power domains for the processing system and programmable logic, allowing independent power management and optimization.

The chip incorporates advanced on-chip interconnect infrastructure that enables high-bandwidth, low-latency communication between the processing system and programmable logic. This interconnect utilizes ARM's Advanced eXtensible Interface (AXI) protocol, providing standardized interfaces for memory-mapped transactions, streaming data, and coherent memory access.

Memory Subsystem Architecture

The ZedBoard platform includes a sophisticated memory hierarchy designed to support both the ARM processor's requirements and the programmable logic's data

processing needs. The primary system memory consists of 512 MB of DDR3 SDRAM operating at 533 MHz, providing 4.26 GB/s of peak memory bandwidth through a 32-bit interface.

The memory controller, integrated within the processing system, features advanced capabilities including error correction coding (ECC), quality of service (QoS) arbitration, and support for various DDR3 configurations. The controller can simultaneously handle memory requests from multiple sources including the ARM cores, DMA controllers, and programmable logic masters.

On-chip memory resources include 256 KB of on-chip RAM (OCM) that can be accessed by both the processing system and programmable logic with single-cycle latency. This OCM serves as high-speed scratch memory for time-critical operations and can be configured as tightly coupled memory for the ARM cores or as shared memory for inter-processor communication.

Fig 3. 16 Memory System Architecture

Processing System Architecture

ARM Cortex-A9 Dual-Core Implementation

The processing system centers around ARM's Cortex-A9 MP Core processor, implemented as a dual-core symmetric multiprocessing (SMP) configuration. Each core

operates at frequencies up to 866 MHz and includes a comprehensive set of ARM architectural features including NEON SIMD extensions, VFPv3 floating-point unit, and ARM TrustZone security extensions.

The Cortex-A9 implementation includes 32 KB of instruction cache and 32 KB of data cache per core, both featuring 4-way set-associative organization with write-back, write-allocate policies. The shared L2 cache provides 512 KB of capacity with 8-way set-associativity, supporting both cores and external masters through the ARM PL310 cache controller.

Snoop Control Unit and Cache Coherency

The ARM SCU (Snoop Control Unit) maintains cache coherency between the dual cores through hardware-based coherency protocols. The SCU monitors memory transactions and ensures that cached data remains consistent across both processor cores, enabling efficient SMP operation without software intervention.

The coherency mechanism extends beyond the processor cores to include external masters in the programmable logic through the Accelerator Coherency Port (ACP). This allows hardware accelerators implemented in the FPGA fabric to participate in the cache coherency protocol, accessing cached data directly without requiring explicit cache management operations.

Fig 3. 17 ARM Cortex-A9 Dual-Core Architecture

Integrated Peripherals and I/O

The processing system incorporates a comprehensive set of integrated peripherals that reduce the need for external components and simplify system design. Key peripherals include dual Gigabit Ethernet MACs with IEEE 1588 precision time protocol support, dual USB 2.0 controllers with on-chip PHYs, dual CAN 2.0B controllers, and multiple UART, SPI, and I2C interfaces.

The GPIO controllers provide 54 pins of general-purpose I/O with programmable drive strength, slew rate control, and pull-up/pull-down configuration. The GPIO system integrates with the interrupt controller to provide efficient event handling for external signals. Storage interfaces include dual SD/SDIO/MMC controllers supporting SD 3.0 specifications, NAND flash controller with hardware ECC generation and correction, and Quad-SPI controller for serial NOR flash memory. These interfaces enable flexible boot configurations and support for various storage media.

Fig 3. 18 Integrated Peripherals and I/O

Programmable Logic Integration

FPGA Fabric Architecture

The programmable logic section of the Zynq-7020 provides 85,000 logic cells organized in a modern FPGA architecture featuring 6-input lookup tables (LUTs), distributed RAM, shift registers, and dedicated carry logic. The fabric includes 560 KB of block RAM configured as 36 Kb blocks that can be cascaded or subdivided to support various memory configurations.

Digital signal processing capabilities are enhanced through 220 DSP48E1 slices, each capable of performing 25×18-bit multiplication with 48-bit accumulation at frequencies exceeding 600 MHz. These DSP slices support various arithmetic operations including multiplication, multiply-accumulate, three-input addition, and barrel shifting.

The clock management system provides eight mixed-mode clock managers (MMCMs) and four phase-locked loops (PLLs) for flexible clock generation and distribution. The clock network features global, regional, and I/O clock routing resources optimized for low skew and high-frequency operation.

AXI Interconnect Infrastructure

The interface between processing system and programmable logic utilizes ARM's Advanced eXtensible Interface (AXI) protocol, implemented through multiple high-performance interconnect ports. The AXI interconnect supports three protocol variants: AXI4 for memory-mapped transactions, AXI4-Lite for simple control registers, and AXI4-Stream for high-throughput streaming data.

Four high-performance AXI ports provide 32-bit or 64-bit interfaces operating at frequencies up to 150 MHz, delivering aggregate bandwidth exceeding 9.6 GB/s between processing system and programmable logic. These ports support advanced features including write response reordering, read data interleaving, and quality of service arbitration.

The General Purpose AXI ports offer additional connectivity for control and status registers, enabling the ARM processors to configure and monitor hardware accelerators implemented in the programmable logic. These 32-bit interfaces operate at up to 250 MHz and support both AXI4 and AXI4-Lite protocols.

Fig 3. 19 Zynq-7020 SoC Architecture

3.3.3 Software Defined Radio Architecture

The diagram illustrates the functional architecture of a Software Defined Radio (SDR) system, encompassing both hardware and software components essential for flexible and reconfigurable wireless communication.

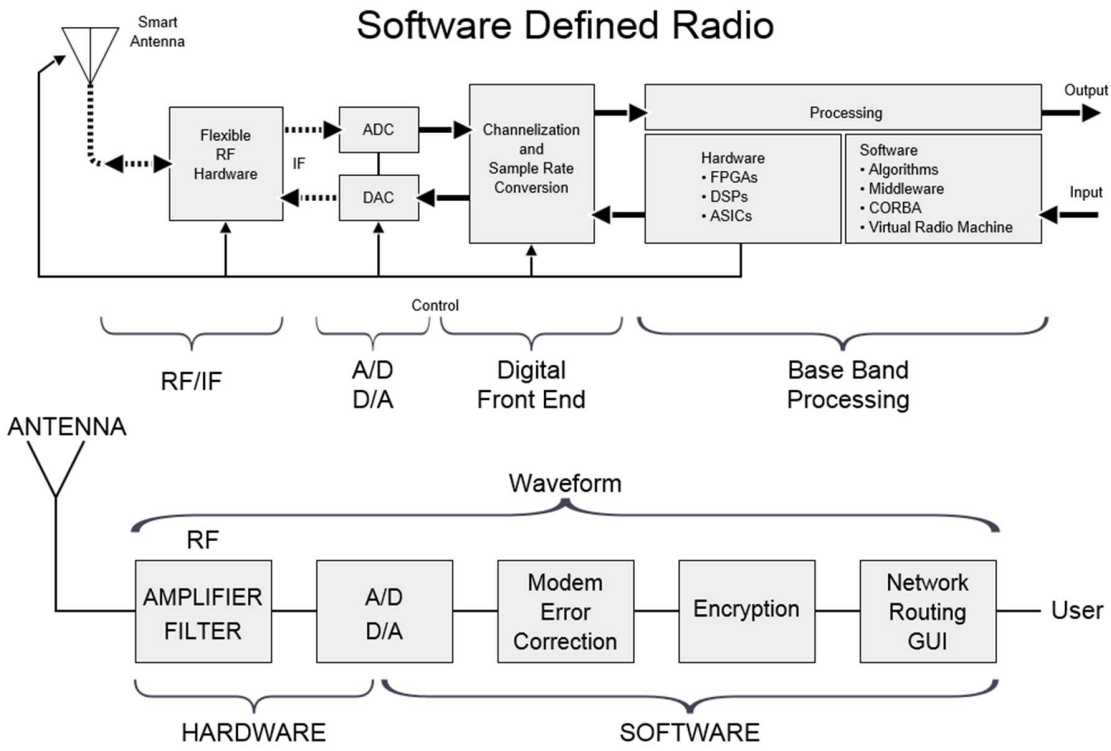


Fig 3. 20 Typical SDR Architecture

System Components

1. Antenna

The antenna serves as the interface between the SDR system and the wireless environment, facilitating the transmission and reception of electromagnetic signals. In advanced configurations, smart antennas are employed to enhance signal quality through beamforming and spatial filtering techniques.

2. RF/IF Front-End

This stage comprises reconfigurable radio frequency hardware that downconverts received RF signals to an intermediate frequency (IF) and vice versa. Its adaptability enables support for multiple frequency bands and modulation schemes.

3. Analog-to-Digital and Digital-to-Analog Conversion

The analog-to-digital converter (ADC) digitizes incoming analog signals, enabling subsequent digital processing. Conversely, the digital-to-analog converter (DAC) transforms processed digital data into analog form for transmission. These converters form the bridge between the analog RF domain and the digital processing chain.

4. Digital Front-End

This module performs critical signal conditioning tasks such as channelization and sample rate conversion. Channelization separates individual communication channels from a composite signal, while sample rate conversion aligns the data rate with the processing system's requirements.

5. Baseband Processing

Baseband processing is implemented using programmable hardware platforms such as FPGAs, DSPs, or ASICs, and is governed by software-defined algorithms. The software layer includes signal processing routines, middleware frameworks, and, in some cases, virtualization environments (e.g., virtual radio machines) to manage dynamic reconfiguration and interoperability.

Waveform Processing Chain

A detailed signal flow is depicted in the lower section of the diagram, emphasizing the interplay between hardware and software in the waveform processing pipeline:

- **Amplification and Filtering**: The RF signal is amplified and filtered to improve signal integrity and suppress interference.
- **Data Conversion**: Digitization of the signal enables precise and flexible processing.
- **Modem and Error Correction**: Modulation and demodulation operations are performed, along with error correction coding to enhance data reliability.
- **Encryption**: Data is secured using cryptographic techniques to ensure confidentiality and integrity.
- **Networking and User Interface**: Processed data is routed through networking layers and presented via a graphical user interface for user interaction and control.

Implementation using FMCOMMS2 and Zynq Boards

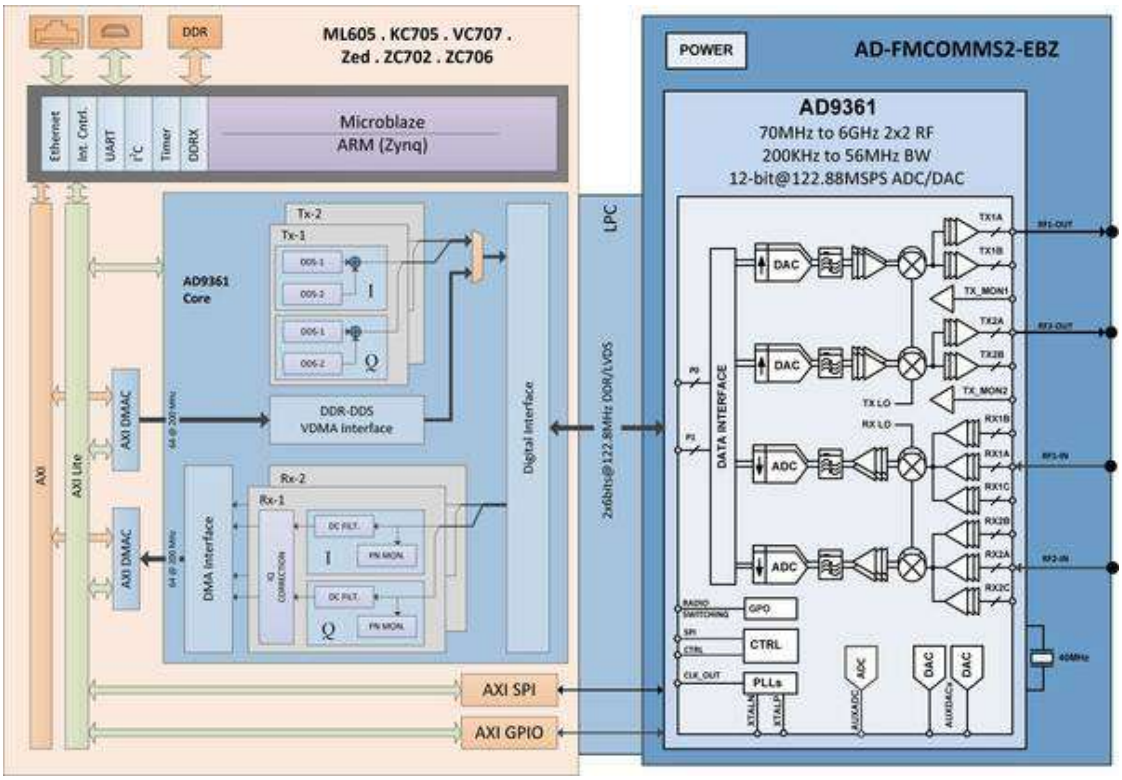


Fig 3. 21 Block Diagram of Zynq Boards + FMCOMMS2

Functional Overview of AD9361-Based SDR System

The block diagram illustrates the integration of the AD9361 transceiver with FPGA-based processing platforms to form a complete Software Defined Radio (SDR) system. The architecture supports high-speed, full-duplex RF signal transmission and reception across a wide frequency range and bandwidth.

AD9361 RF Front-End (Right Side: AD-FMCOMMS2-EBZ Module)

The **AD9361** is a highly integrated RF agile transceiver that operates from 70 MHz to 6 GHz, supporting channel bandwidths ranging from 200 kHz to 56 MHz. The transceiver features:

- **Dual Transmit (Tx) and Receive (Rx) Chains**: Each path includes 12-bit DACs and ADCs operating at up to 122.88 MSPS, enabling wideband I/Q sampling.

- **Programmable Gain Stages and Mixers**: Allow precise tuning of the RF signal chain for both upconversion and downconversion.
- **Digital Interface**: Connects the transceiver to the baseband processing hardware over a **LVDS (Low Voltage Differential Signaling)** interface.
- **Integrated PLLs and Filters**: Facilitate frequency synthesis and channel selection, minimizing external component requirements.

FPGA/SoC Baseband Processing Platform (Left Side)

This section represents the programmable logic and processing system (e.g., Xilinx Zynq, MicroBlaze, or other FPGA platforms), which manages signal processing tasks and system control:

- **Tx and Rx Processing Cores**: Handle digital upconversion (DUC), downconversion (DDC), and interpolation/decimation. The Tx cores include Direct Digital Synthesizers (DDS) and interpolation filters for waveform generation. Rx chains perform decimation, digital filtering, and I/Q separation.
- **VDMA (Video Direct Memory Access)**: Facilitates high-speed data transfer between the memory and the processing cores, ensuring real-time throughput for streaming applications.
- **AXI Interfaces**: Utilize Advanced eXtensible Interface (AXI) protocols to connect system components, including AXI SPI for transceiver configuration and AXI GPIO for general control tasks.
- **Microprocessor Subsystem**: An embedded processor (ARM or MicroBlaze) provides software control, enabling configuration, monitoring, and higher-layer protocol management.

System Integration and Data Flow

Data captured by the receive path is digitized by the AD9361 and transferred via the digital interface to the FPGA for baseband processing. Processed data may be routed to memory or passed to higher layers for application use. For transmission, the inverse process

is followed—baseband data is synthesized, interpolated, and sent to the DACs for RF conversion and transmission via the antenna.

This tight integration of high-speed RF front-end and programmable baseband processing enables flexible, reconfigurable radio platforms suitable for modern wireless communication systems.

4

SYSTEM BUILD AND DEPLOYMENT

4.System Build and Deployment

4.1 Hardware Platform

The choice of the Zynq Zedboard and AD9361 RF frontend was driven primarily by availability constraints within our educational institution. The Zynq Zedboard integrates a Xilinx Zynq-7000 SoC, which combines a dual-core ARM Cortex-A9 processor with programmable logic (FPGA fabric), providing an optimal balance between software programmability and hardware acceleration. This architecture is particularly well-suited for real-time signal processing tasks, such as those encountered in wireless communication systems.

The AD9361 is a highly flexible RF transceiver that supports a wide frequency range (70 MHz to 6 GHz), tunable bandwidth (up to 56 MHz), and both transmit and receive paths, making it ideal for prototyping various wireless standards. It integrates seamlessly with the Zynq platform via the FMC connector, and is supported by robust open-source tools and drivers from Analog Devices and Xilinx.

Although higher-end alternatives like the ZC706 board offer additional features and performance, the Zedboard was selected due to accessibility and cost constraints without significantly compromising capability. Together, the AD9361 and Zynq Zedboard provide a powerful and cost-effective SDR platform that supports end-to-end physical layer development, ideal for academic research and prototyping in wireless communication.

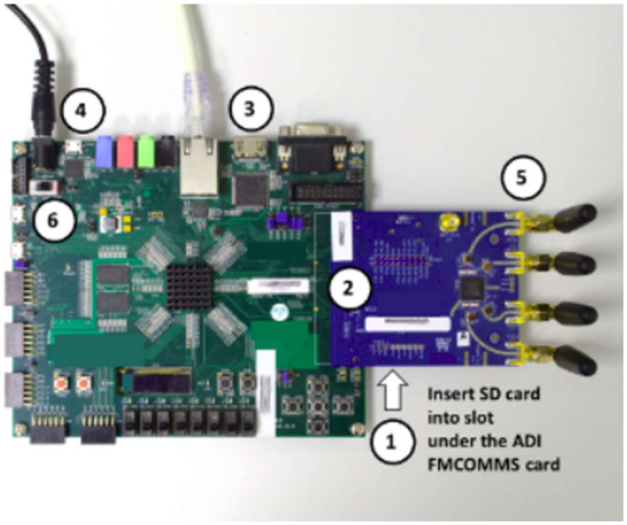


Fig 3. 22 Zedboard + AD9361

4.2 FPGA Design and HDL Integration

4.2.1 FPGA Hardware Design and Integration using Vivado

The foundation of the SDR system is built in Xilinx Vivado using the IP Integrator flow. The ZYNQ7 Processing System IP block is configured to instantiate the dual-core ARM Cortex-A9 processor along with its associated PS peripherals. In the block diagram. this PS block interfaces with multiple IPs through AXI-Lite for control and AXI-Stream or direct connections for data flow.

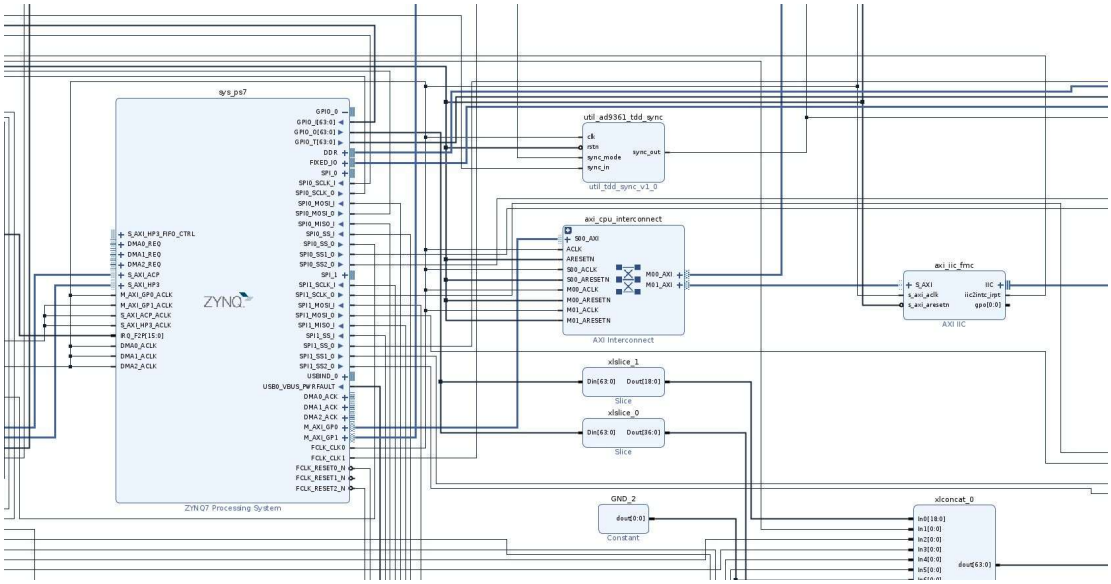


Fig 4. 1 Zynq–AD9361 System Integration

Key components in the PL (Programmable Logic) include the axi_ad9361 core, which serves as the bridge between the digital logic and the AD9361 RF transceiver. It handles bidirectional I/Q data transfer and configuration via the SPI interface. Surrounding it, the axi_ad9361_dac_fifo ensures smooth transmission by buffering DAC samples, while util_ad9361_tdd_sync ensures TDD synchronization between transmission and reception.

Interfacing and system control are achieved using additional IPs like axi_iic, used to configure peripheral devices (such as clock generators or EEPROMs) over I2C, and axi_gpio to manage reset lines or other control signals. These IPs are connected to the PS via AXI

interconnects. The entire design is synthesized, implemented, and exported to SDK or Vitis, where software is written to initialize and control the RF system.

4.2.2 Signal Path through AD9361 and FPGA – RF to Baseband

The signal chain between the analog RF domain and digital baseband processing is shown in a detailed block diagram. This diagram breaks down both the receive and transmit chains across the AD9361 and FPGA boundary.

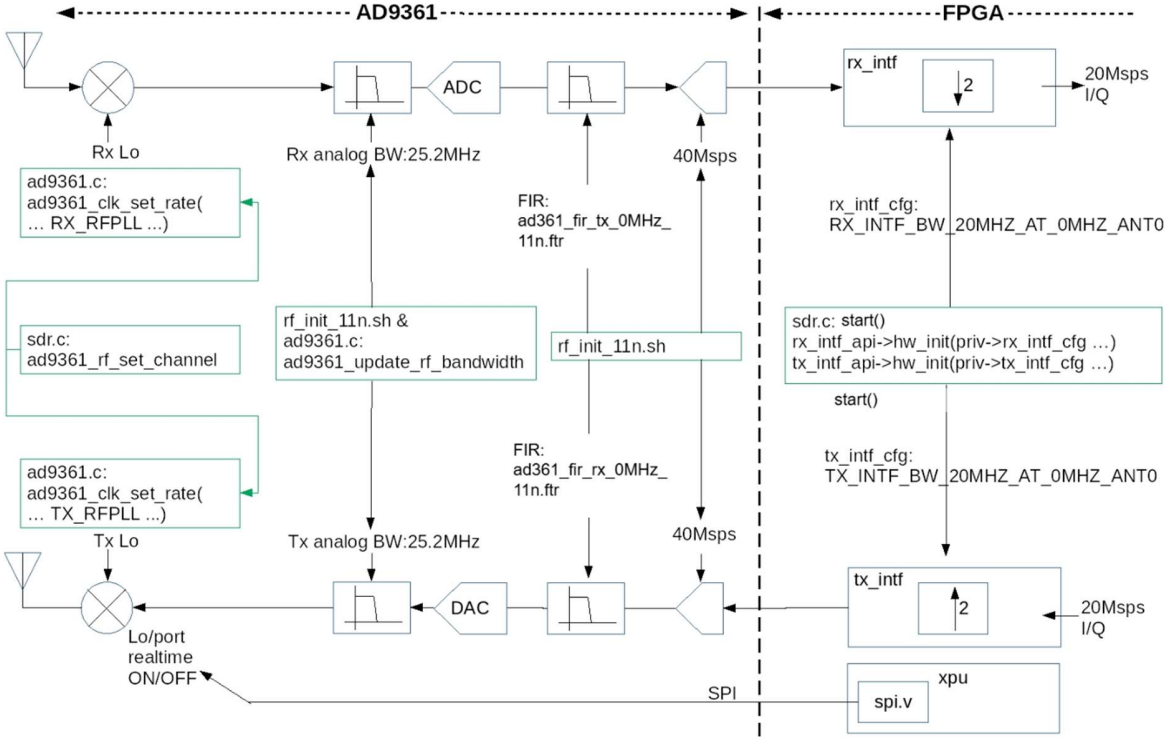


Fig 4. 2 RF-to-Baseband Chain

On the receive path, the antenna signal is downconverted using the RX LO inside the AD9361. The LO is programmed via SPI using the function ad9361_clk_set_rate(... RX_RPLL ...). The analog signal is filtered and digitized using the onboard ADC at 40 MSps. A programmable FIR filter is then applied inside the AD9361 (loaded via SPI using .ftr files like fir_rx_0MHz_11n.ftr). The resulting I/Q samples are transmitted to the FPGA.

Inside the FPGA, the samples enter the rx_intf block which decimates them to 20 MSps. The decimated samples are passed on to either the processor system or stored for further processing. Runtime configuration for bandwidth, LO, and sampling behavior is handled via functions like ad9361_update_rf_bandwidth and initialization scripts like rf_init_11n.sh.

On the transmit side, I/Q data is generated in the baseband, typically at 20 MSps. The tx_intf block inside the FPGA interpolates this to 40 MSps, after which the samples are pushed to the AD9361's transmit path. The samples are filtered using a programmable FIR (e.g., fir_tx_0MHz_11n.ftr), converted to analog by the DAC, and upconverted to RF using the TX LO (configured with ad9361_clk_set_rate(... TX_RPLL ...)).

Both TX and RX interfaces are initialized during runtime by calling hardware initialization APIs (rx_intf_api->hw_init(...) and tx_intf_api->hw_init(...)). The analog bandwidth is typically set to 25.2 MHz to match the 20 MHz baseband bandwidth used in many communication protocols.

4.2.3 Software Interface and AD9361 Runtime Configuration

The software layer running on the ARM processor of the Zynq SoC orchestrates the overall system. This includes initializing the AD9361, programming its registers via SPI, and dynamically adjusting parameters like LO frequency, bandwidth, and filter coefficients.

The SPI interface is implemented in the PL using spi.v, which communicates with the AD9361 to execute commands like ad9361_rf_set_channel, ad9361_update_rf_bandwidth, and clock configuration functions. These functions are typically called from higher-level scripts (e.g., rf_init_11n.sh) or embedded C programs running within the system.

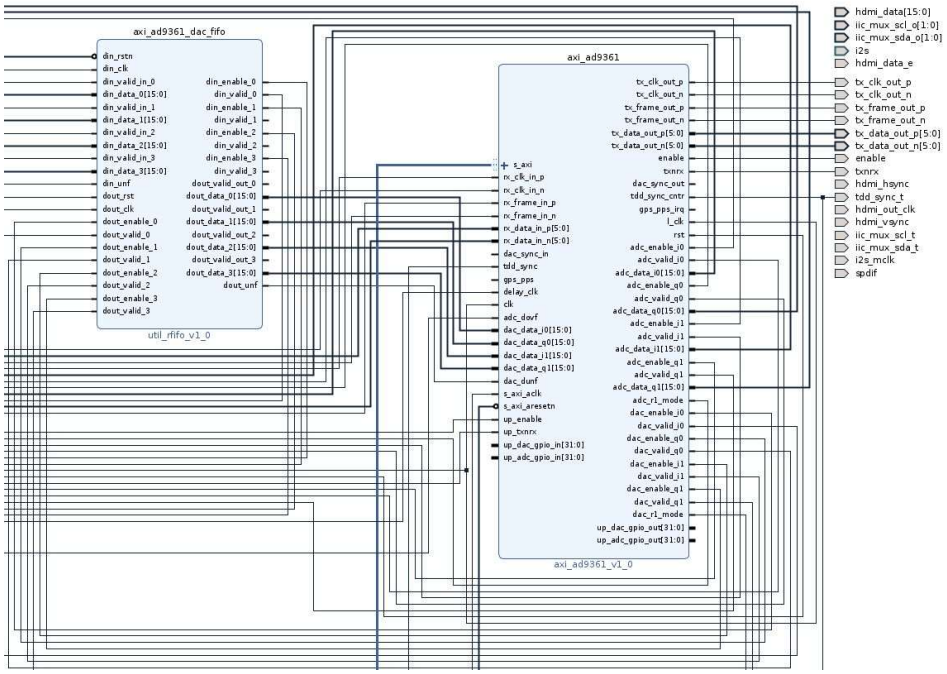


Fig 4. 3 Digital IF & Config Flow

Runtime structures such as rx_intf_cfg and tx_intf_cfg hold the configuration for each path, including center frequency, antenna selection, and data rate. These are passed into initialization APIs that configure the FPGA logic for real-time streaming.

The FPGA's AXI4-Lite interface enables low-latency communication between software and programmable logic. The Linux-based runtime software stack or bare-metal firmware can directly write to registers within axi_ad9361, configure GPIOs for hardware control, and monitor status flags for signal quality, lock status, or TDD sync.

This software-hardware co-design is essential for building an agile SDR platform. The design allows users to implement various communication protocols and modify RF parameters on the fly without rebuilding the hardware design.

4.3 Software System and Control

The build flow diagram outlines the complete process of developing a bootable SD card image for a Zynq-based embedded system utilizing the AD9361 RF transceiver. This

design flow is critical for establishing a software-defined radio (SDR) platform capable of executing real-time wireless communication tasks. The process ensures tight integration between custom programmable logic, the embedded processing system, and a Linux-based operating environment.

The design process begins in Vivado Design Suite, where the hardware description language (HDL) design is developed. This design includes the instantiation of IP cores such as the AD9361 interface (provided by Analog Devices), AXI interconnects, clocking resources, and other peripheral components necessary for system functionality. Once the design is synthesized, implemented, and verified, a bitstream (.bit file) is generated. This file configures the programmable logic (PL) portion of the Zynq SoC at boot time.

Concurrently, the U-Boot bootloader is compiled and configured. U-Boot is responsible for initializing the processing system (PS), loading the Linux kernel and device tree, and handing over control during the boot sequence. It includes environment variables and boot scripts tailored for Zynq platforms.

The Linux kernel is then cross-compiled for the ARM Cortex-A9 processor in the Zedboard. This kernel includes device drivers for the AD9361 RF transceiver, enabling baseband I/Q data interfacing, configuration via SPI, and digital communication through the programmable logic. Additionally, network drivers and interfaces (e.g., Ethernet, USB) are enabled to allow remote access and data transmission.

To support user-space operations, an Ubuntu root filesystem (RootFS) is either built or customized. This filesystem contains essential Linux utilities, networking tools, development packages, and libraries required to operate the SDR platform and run high-level applications. The root filesystem is typically created using tools such as debootstrap or Buildroot and may include additional software for SDR control and data logging.

In the Image Assembly phase, all individual components are combined into a unified bootable image. This includes:

- The U-Boot bootloader
- The Linux kernel image (zImage or uImage)
- The Device Tree Blob (DTB), which describes the hardware layout to the kernel
- The FPGA bitstream
- The Ubuntu RootFS

These components are integrated using build scripts and tools such as petalinux-build, bootgen, or custom Makefiles. The final output is a single SD card image (.img file) that contains boot and root partitions, ready to be flashed onto a microSD card.

Upon powering the Zynq Zedboard with the SD card inserted, the bootloader initiates the system, loads the bitstream to configure the FPGA, and then boots the Linux kernel. The system transitions into a fully functional embedded Linux environment with RF transceiver support, network connectivity, and the ability to run user-defined applications for wireless communication experiments.

This build flow is essential in academic and prototyping environments, where flexibility, customization, and low-level control over hardware and software are required. It provides a robust foundation for SDR development, enabling rapid testing and deployment of communication protocols and signal processing algorithms on real hardware.

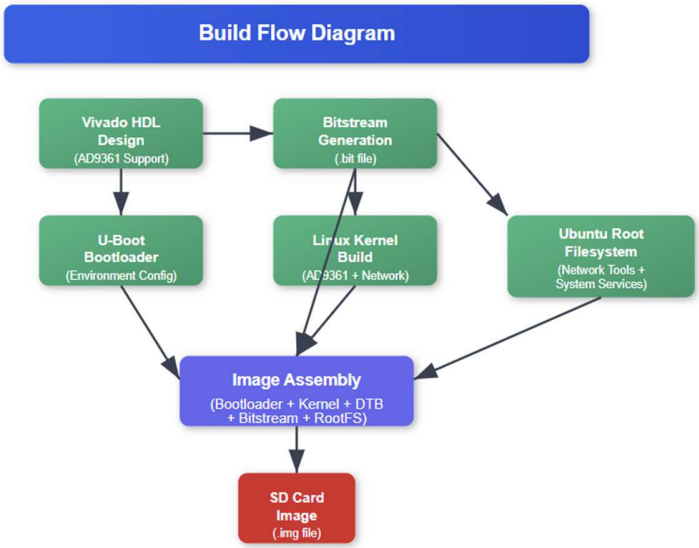


Fig 4. 4 Build Flow Diagram

4.4 System Image Preparation and Boot Process

The system setup for the Zynq-based SDR platform involves a structured build and boot process, beginning with the generation of hardware and software components and culminating in a fully operational Linux environment with AD9361 RF support. The entire workflow, from bitstream generation to Linux boot-up, is illustrated in the accompanying Build Flow Diagram and Boot Sequence Diagram.

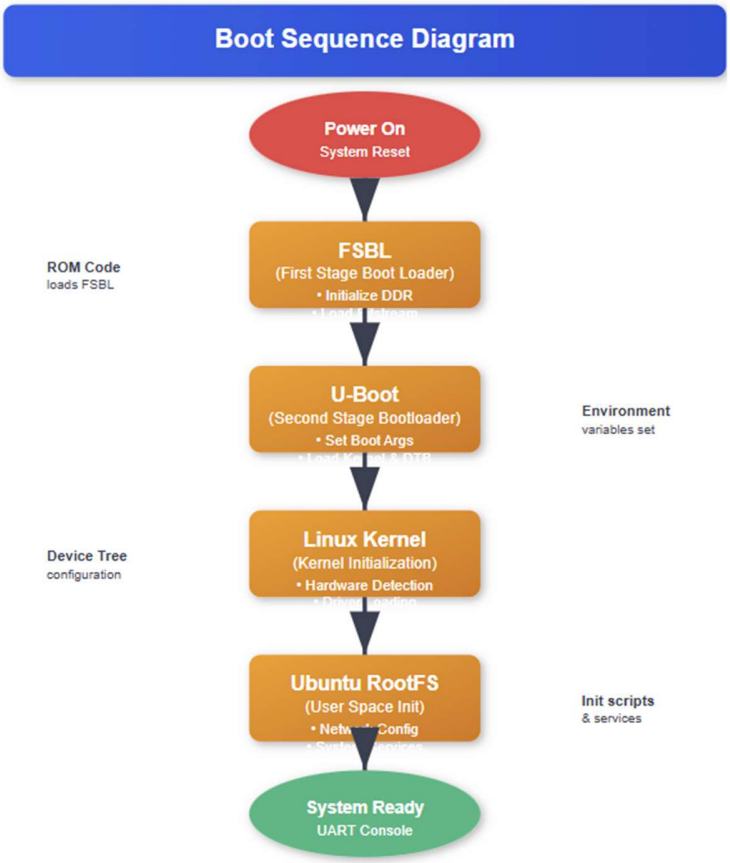


Fig 4. 5 Boot Sequence Diagram

The process begins with hardware design in the Vivado Design Suite, where custom HDL is created to interface with the AD9361 RF transceiver and other peripherals. Upon successful synthesis and implementation, a .bit file is generated. This bitstream configures the programmable logic (PL) of the Zynq SoC and is essential for the hardware's runtime operation.

Subsequently, the U-Boot bootloader is built and configured for the target hardware. As the second-stage bootloader, U-Boot is responsible for initializing system parameters, loading the kernel and device tree, and passing necessary boot arguments. Environment variables such as memory mappings, console settings, and root filesystem locations are also defined during this phase.

Parallel to the bootloader setup, the Linux kernel is cross-compiled with support for the AD9361 transceiver, network interfaces, and system control features. This ensures that once the kernel boots, it can properly initialize and interact with the underlying hardware as specified in the device tree.

A minimal Ubuntu-based root filesystem is then created. This rootfs includes essential user-space utilities, libraries, networking tools, and SDR applications. It serves as the runtime environment once the kernel is initialized.

With all components prepared, they are assembled into a single bootable SD card image. This image includes the U-Boot bootloader, Linux kernel image, device tree blob (DTB), FPGA bitstream, and the Ubuntu root filesystem. Image assembly tools or manual scripting techniques are used to package these into a bootable .img file. The image is then flashed onto an SD card using tools such as dd or balenaEtcher.

When the system is powered on, the boot process begins, as depicted in the Boot Sequence Diagram. The on-chip ROM first loads the First Stage Bootloader (FSBL), which initializes the DDR memory and programs the FPGA with the generated bitstream. Control is then passed to U-Boot, which sets the boot arguments and loads both the kernel and device tree.

Next, the Linux kernel is initialized, detecting hardware components based on the device tree configuration and loading appropriate drivers. Once the kernel completes its

setup, control is handed over to the Ubuntu root filesystem, which executes user-space initialization scripts, configures network interfaces, and starts necessary system services.

5

RESULTS AND ANALYSIS

5.Result and Analysis

The testing environment for evaluating the software-defined radio (SDR)-based wireless system was meticulously designed to ensure accurate performance assessment and signal integrity. The setup comprised the following major components: a client system (host PC), an SDR router platform (Zynq ZedBoard with AD9361), network connectivity via Ethernet or Wi-Fi, and diagnostic instruments such as a spectrum analyzer to verify RF characteristics.

The client system used for testing was a high-performance Linux-based PC equipped with necessary development and communication tools. This system was responsible for transmitting data to and receiving data from the SDR router. It also facilitated firmware deployment, kernel logging, and interaction with user-space applications. SSH access was enabled to ensure real-time diagnostics and remote command execution.

The SDR router was implemented using a Xilinx Zynq ZedBoard integrated with the Analog Devices AD9361 RF front-end. The ZedBoard acted as the central wireless processing unit. It handled baseband signal processing in the programmable logic (PL) and network stack management in the ARM processing system (PS). The AD9361 enabled full-duplex transmission and reception within the 2.4 GHz ISM band, with programmable gain control, bandwidth adjustment, and tunable center frequency. This made it highly suitable for dynamic spectrum experiments and custom MAC/PHY layer testing.

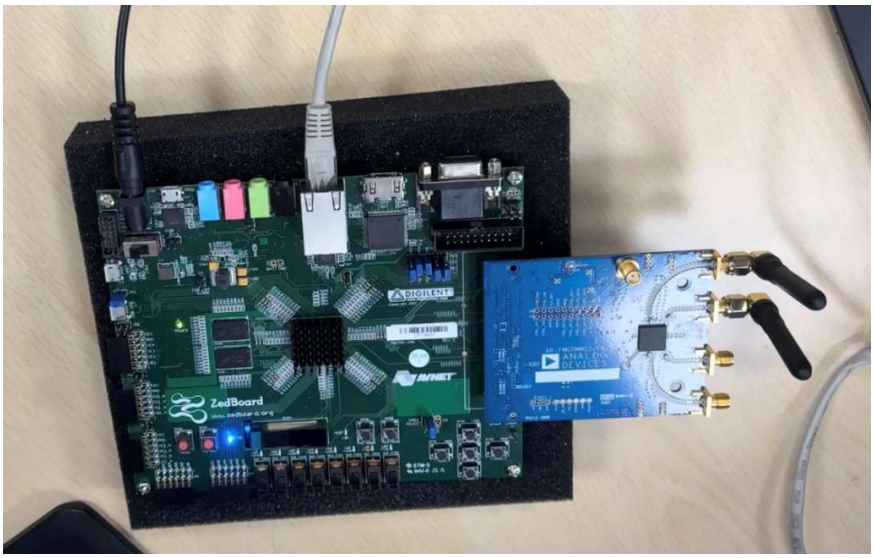


Fig 5. 1 SDR Integration Setup

The ZedBoard was connected to the client system using Gigabit Ethernet to provide stable and low-latency communication. In scenarios simulating end-user environments, Wi-Fi connectivity was also employed. For this, the stack running on the SDR acted as a wireless access point, and the client system operated as a Wi-Fi station. This setup allowed for realistic over-the-air (OTA) performance measurements under varying signal conditions and modulation configurations.

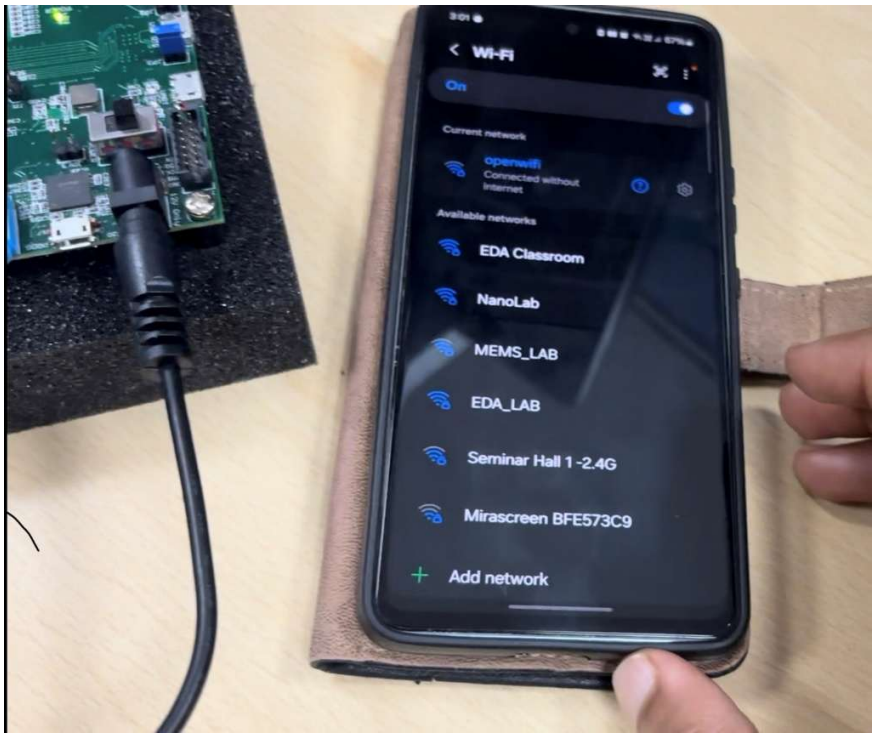


Fig 5. 2 SSID in Mobile

For system diagnostics and RF validation, a spectrum analyzer was employed to monitor the real-time transmission characteristics of the SDR. The analyzer was connected via coaxial or directional coupler to the antenna output of the AD9361. It provided crucial insights into spectral occupancy, spurious emissions, frequency stability, and effective radiated power (ERP). As shown in the attached spectrum plot, the observed transmission is centered around 2.4 GHz, with a well-defined peak corresponding to the modulated signal, surrounded by side lobes indicative of OFDM or similar multi-carrier schemes. The noise

floor and adjacent channel rejection performance are also evident from the trace, validating the system's spectral efficiency and filter design.

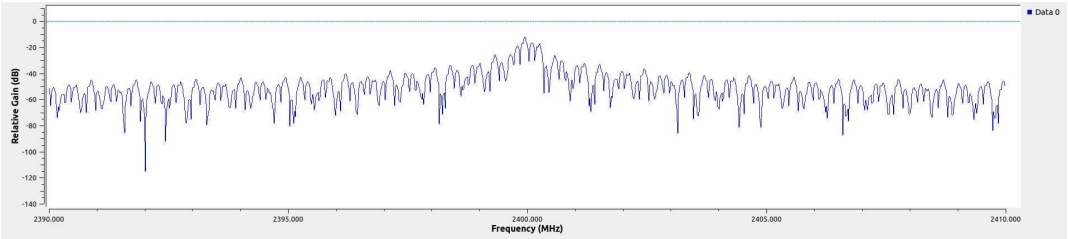


Fig 5. 3 Spectrum Analyser Output

Overall, the test setup provided a controlled, flexible, and observable environment for evaluating SDR-based wireless communication systems. It effectively balanced between hardware-level signal verification and software-level protocol testing, enabling comprehensive performance benchmarking and iterative development.

6

CONCLUSION AND FUTURE SCOPE

6.Conclusion and Future Scope

6.1 Conclusion

This project presents the end-to-end design and implementation of a fully functional IEEE 802.11-based wireless router, built entirely using a custom hardware and software stack. The objective was to create a reliable and efficient system capable of wireless packet transmission and routing without relying on any pre-existing frameworks or third-party solutions. Every layer—from physical signal processing to network-level routing—was designed, implemented, and validated from the ground up.

At the hardware level, a modular FPGA-based baseband architecture was developed to perform essential wireless communication tasks such as modulation, demodulation, synchronization, and channel estimation. This was integrated with a custom-designed MAC layer, responsible for managing access control and enabling efficient medium sharing. The entire hardware design was optimized for timing and resource efficiency to ensure robust operation on the target platform.

On the software side, a lightweight embedded Linux-based stack was implemented. This included custom device drivers, initialization scripts, and core routing functionalities to enable seamless data flow between wireless and Ethernet interfaces. Special emphasis was placed on maintaining stable and consistent routing performance under varying network loads.

A key focus of the project was the integration of hardware and software components, requiring a thorough understanding of digital systems, embedded programming, and network protocols. Through systematic testing and refinement, the design evolved into a robust system capable of real-time operation.

This project offered deep technical exposure across the entire system stack—from HDL development and FPGA integration to Linux networking and device driver development. It demonstrates the feasibility of developing a complete wireless networking

system using IEEE 802.11 standards and provides a flexible foundation for future enhancements and feature extensions. This work establishes a strong base for continued exploration in wireless technologies, embedded systems, and network engineering.

6.2 Future Scope

The custom wireless router developed in this project presents a flexible foundation for several advanced applications, particularly in smart transportation and energy-efficient communication systems.

One major future direction is the adaptation of this platform for Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication. By optimizing the communication stack for low latency and high reliability, this router can serve as a node in an intelligent transport system. It could enable vehicles to exchange real-time data such as speed, location, or road hazards, and communicate with infrastructure like traffic lights or road-side units. This would improve road safety, reduce traffic congestion, and support autonomous driving technologies. Since the hardware and software are fully under our control, protocol modifications — such as priority-based packet scheduling and time-sensitive delivery — can be implemented efficiently to meet the demands of high-mobility environments.

Fig 6. 1 V2V and VTI Infrastructure

Another promising direction is optimizing the router for low-power Wi-Fi variants, particularly suited for IoT applications. Many IoT devices, such as environmental sensors, home automation systems, and wearable health monitors, operate under strict energy constraints. By modifying the MAC layer and introducing sleep scheduling, low-duty-cycle operation, and reduced overhead in packet headers, this router can serve as a base station for energy-efficient wireless communication. This opens up use cases in smart homes, agriculture, industry 4.0, and health monitoring systems where long battery life and reliable connectivity are essential.

The router can also be adapted for edge computing by embedding lightweight processing modules. This would allow local processing of data from IoT or vehicular systems, reducing reliance on cloud infrastructure and lowering response times.

In summary, the developed wireless router has strong potential for application in emerging fields such as intelligent transport systems and IoT networks. Its open and customizable architecture makes it suitable for rapid prototyping, deployment, and optimization in real-world smart environments.

REFERENCES

[1] Cho, H. H., Lai, C. F., Shih, T. K., & Chao, H. C. (2014). Integration of SDR and SDN for 5G. IEEE Access, 2, 1196–1204. https://doi.org/10.1109/ACCESS.2014.2357435

[2] ZedBoard Zynq^{TM} Evaluation and Development Hardware User's Guide. (2012).

[3] Aslam, M., Jiao, X., Liu, W., Mehari, M., Havinga, T., & Moerman, I. (2024). A novel hardware efficient design for IEEE 802.11ax compliant OFDMA transceiver. Computer Communications, 219, 173–181. https://doi.org/10.1016/j.comcom.2024.03.006

[4] Restuccia, F., & Melodia, T. (2019). Big Data Goes Small: Real-Time Spectrum-Driven Embedded Wireless Networking Through Deep Learning in the RF Loop. http://arxiv.org/abs/1903.05460

[5] Abidi, A. A. (2004). RF CMOS Comes of Age. IEEE Journal of Solid-State Circuits, 39(4), 549–561. https://doi.org/10.1109/JSSC.2004.825247

[6] ad9361. (n.d.).

[7] 2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). (2020). IEEE.

[8] Akeela, R., & Dezfouli, B. (2018). Software-defined Radios: Architecture, state-of-the-art, and challenges. In Computer Communications (Vol. 128, pp. 106–125). Elsevier B.V. https://doi.org/10.1016/j.comcom.2018.07.012

[9] Zanna, P., Kumar, D., & Radcliffe, P. J. (2021). A novel method for utilizing RF information from IEEE 802.11 frames in Software Defined Networks. MethodsX, 8. https://doi.org/10.1016/j.mex.2021.101544

[10] Zitouni, R., Bouaroua, H., & Senouci, B. (2020). Hardware-Software Codesign for Software Defined Radio: IEEE 802.11p receiver case study. http://arxiv.org/abs/2003.09525

[11] Accelerating FPGA-Based Wi-Fi Transceiver Design and Prototyping by High-Level Synthesis. (n.d.).

[12] Mehraeen, S., & Rahimi, R. (2024). Design and implementation of SDR (software-defined radio) systems for enhanced signal processing: An experimental study. International Journal of Communication and Information Technology, 5(1), 43–46. https://doi.org/10.33545/2707661x.2024.v5.i1a.80

[13] Sun, W., Wu, Q., & Xiang, Y. (2011). Design and implementation of the FPGA-based wireless network card. Procedia Engineering, 15, 2234–2238. https://doi.org/10.1016/j.proeng.2011.08.418